

Trabajo Fin de Grado

CONTROL DE UN PUENTE GRÚA DIDÁCTICO MEDIANTE VISUALIZACIÓN DE SU CARGA CONTROL OF A DIDACTIC 3D-CRANE THROUGH VISUALIZATION OF ITS LOAD

Autor/es

Daniel Redondo Sanz

Director/es

José Ramón Beltrán Blázquez

Resumen del Trabajo de Fin de Grado

El presente Trabajo de Fin de Grado consiste en la construcción de un sistema de control de un puente grúa didáctico, con realimentación mediante visualización de su carga. Para conseguir el objetivo, se desarrolla un programa de visión artificial para detectar la posición de la carga del puente grúa y calcular sus coordenadas respecto al sistema de referencia del campo de trabajo del puente grúa. Posteriormente, se analizan los sistemas de control que suministra el fabricante y se elige el más adecuado. El sistema de control elegido se modifica para adaptar los parámetros de los controladores al objetivo del proyecto, y para introducir las coordenadas de la posición de la carga obtenidas con el programa anterior a la realimentación del sistema de control elegido. Por último, se construye un sistema de comunicación entre el programa de visión artificial y el sistema de control del puente grúa con el objetivo de transmitir las coordenadas de la posición de la carga del puente grúa de forma correcta y precisa.

Final Degree Project Summary

Present Final Degree Project consists in the construction of a didactical 3D crane control system, with feedback through artificial vision of its load. For achieve this goal, an artificial vision program is developed in order to detect the 3D crane load position and compute its coordinates respect of the 3D crane workspace reference system. Later, the control systems supplied for the manufacturer are analyzed, and the most suitable control system is chosen. The chosen control system is modified in order to adjust the parameters of the controllers to the project objective, and for introduce of the obtain position coordinates of the load with the previous program to the feedback of the chosen control system. By last, a communication system between the artificial vision program and the 3D crane control system is built in order to transmit correctly and accurately the position coordinates of the 3D crane load.

1. Introducción.....	6
1.1. El puente grúa.....	¡Error! Marcador no definido.
1.2. Sistemas de control.....	7
1.3. Objetivo.....	7
1.4. Estructura del documento	8
2. Contexto del Trabajo de Fin de Grado.....	9
3. Obtención de la posición de la carga por visión	13
3.1. Elementos físicos y programas utilizados	13
3.1.1. Puente grúa didáctico de INTECO	13
3.1.2. Videocámara	15
3.1.3. Matlab	15
3.1.4. Open CV + Python + Spyder (Anaconda).....	15
3.2. Estrategia de visión	16
3.2.1. Dimensiones del espacio de trabajo del puente grúa.....	16
3.2.2. Colocación de las marcas	17
3.2.3. Colocación de la videocámara	17
3.3. Detección de las marcas por videocámara	19
3.3.1. Inicio del programa y grabación de vídeo	19
3.3.2. Obtención de la imagen binaria	20
3.3.3. Filtrado de ruido en la imagen binaria	22
3.4. Obtención de la posición de la carga	23
3.4.1. Obtención de los centros de las marcas de referencia	23
3.4.2. Transformación geométrica de la imagen binaria	26
3.4.3. Obtención del centro de la marca objetivo	27
4. El Puente grúa de INTECO.....	29
4.1. Estructura del puente grúa	29
4.2. Comportamiento del puente grúa	30
4.3. Sistema de control del puente grúa.....	31
4.3.1. Sistemas de control proporcionados	32
4.3.2. Ajuste de los parámetros de los controladores	33
4.4. Comunicación Spyder – Simulink.....	37
5. Resultados	41
5.1. Resultados de rapidez de la aplicación.....	41
5.2. Resultados de precisión de la aplicación	45

6. Conclusiones y propuestas de mejora.....	46
6.1. Conclusiones	46
6.2. Propuestas de mejora.....	46

Índice de anexos

Anexo 1: Programas creados para la aplicación de visión e imágenes obtenidas.

Anexo 2: Características del puente grúa y de sus componentes principales.

Anexo 3: Sistemas de control demo.

Anexo 4: Prueba de parámetros en los controladores del sistema de control.

Anexo 5: Programas de pruebas para la comunicación Spyder-Simulink.

Lista de figuras

Figura 1. Puente grúa real en la industria.	6
Figura 2. Diagrama de bloques de la tesis de control del puente grúa por visión artificial.	10
Figura 3. Esquema de la tesis de modelado y control de un puente grúa.	11
Figura 4. Esquema del objetivo del presente Trabajo de Fin de Grado.	12
Figura 5. Esquema de la relación del hardware necesario para controlar el puente grúa.	14
Figura 6. Imagen del tablero RT-DAC/USB2.	14
Figura 7. Esquema de la conexión de los elementos para la grabación de vídeo.	18
Figura 8. Colocación de las marcas y la videocámara.	18
Figura 9. Imagen original captada por la cámara.	19
Figura 10. Imagen binaria por el método de umbralización.	21
Figura 11. Imagen binaria obtenida por máscara de rojos.	22
Figura 12. Imagen tras filtrado de ruido.	23
Figura 13. Imagen de los centros de las marcas.	24
Figura 14. Imagen de los centros de las marcas de referencia.	26
Figura 15. Imagen en planta obtenida.	27
Figura 16. Captura de pantalla del centro de la marca objetivo.	28
Figura 17. Esquema de los componentes del puente grúa.	29
Figura 18. Relación entre los componentes del puente grúa.	30
Figura 19. Derecha: Motor DC del puente grúa. Izquierda: <i>Encoder</i> del puente grúa.	31
Figura 20. Captura de pantalla de la interfaz gráfica del puente grúa.	32
Figura 21. Captura del sistema de control elegido.	33
Figura 22. Sistema de control modificado para simulación.	34
Figura 23. Señales de posición, oscilación y control en el eje X.	35
Figura 24. Señales de posición, oscilación y control en el eje Y.	35
Figura 25. Señales de posición, oscilación y control en el eje X con PID ajustado.	37
Figura 26. Señales de posición, oscilación y control en el eje Y con PID ajustado.	37
Figura 27. Representación de los datos recibidos a lo largo del tiempo.	38
Figura 28. Primer ejemplo de precisión en la transmisión de datos.	39
Figura 30. Sistema de control modificado.	40
Figura 31. Esquema de los receptores UDP y los osciloscopios en los programas de prueba.	41
Figura 32. Representación OX-t del primer programa de simulación.	42
Figura 33. Representación OY-t del primer programa de simulación.	42

Figura 34. Representación OY-OX del primer programa de simulación.	43
Figura 35. Representación OX-t del segundo programa de simulación.....	44
Figura 36. Representación OY-t del segundo programa de simulación.....	44
Figura 37. Representación OY-OX del segundo programa de simulación.....	45

Lista de tablas

Tabla 1, Parámetros elegidos para los controladores de nuestro sistema de control.	36
---	----

1. Introducción

Una grúa es una máquina que se utiliza para mover cargas. Sus principales movimientos son el desplazamiento de cargas en el plano horizontal y la elevación y descenso de dichas cargas en el eje vertical [1].

Las primeras grúas datan del siglo VI a.C en la Antigua Grecia y fueron desarrolladas posteriormente durante el Imperio Romano, donde se crearon grúas con ventajas mecánicas de 30:1.

Con la llegada de la máquina de vapor de agua de James Watt, en el siglo XVIII se crearon las primeras grúas accionadas por este tipo de máquina. Este suceso supuso una reducción de la mano de obra en el uso de la grúa.

Actualmente, las grúas emplean motores de combustión interna, eléctricos e hidráulicos, aunque también se siguen utilizando las grúas manuales para pequeñas cargas.

1.1. El puente grúa

Un puente grúa es un tipo de grúa móvil que se utiliza en la industria y está principalmente localizada en el interior de naves industriales y fábricas. Se caracteriza por estar montado sobre dos rieles paralelos y situados a una determinada altura en los laterales de una nave. El carril del puente grúa se mueve sobre estos dos rieles y a su vez incorpora un carro que se desplaza sobre ese carril. En este carro se sitúa el sistema de elevación de la carga [2].



Figura 1. Puente grúa real en la industria [22].

1.2. Sistemas de control

Un sistema de control es un sistema que regula el comportamiento de otro sistema con un objetivo determinado.

Un sistema de control en bucle cerrado es un tipo de sistema de control que compara la entrada del sistema a ser controlado con la salida obtenida del mismo, e introduce la diferencia entre la entrada y la salida al controlador del sistema junto con el valor de la entrada, para dar un valor de salida cada vez más preciso y cercano al valor de dicha entrada, disminuyendo el error o diferencia entre los valores de entrada y de salida lo máximo posible.

Con el avance de la ingeniería de control, se han creado sistemas de control aplicados a la familia de las grúas, permitiendo una gran automatización de las mismas, y uno de los problemas que se abordan actualmente en el mundo de la ingeniería es el desarrollo de estos sistemas de control para aumentar la velocidad y precisión de los movimientos de este tipo de máquinas.

1.3. Objetivo

El objetivo de este Trabajo de Fin de Grado es la obtención de un sistema de control en bucle cerrado del puente grúa, con una realimentación de la posición en el plano horizontal de la carga de la grúa mediante su visualización con una videocámara externa.

Con este fin, se obtienen por videocámara las coordenadas de la posición de la carga del puente grúa en el plano horizontal (Ejes X e Y) y se transfieren al sistema de control, donde se comparan con las coordenadas de la posición deseada de la carga y, en función de su diferencia, el controlador actúa sobre los motores que alimentan el puente grúa para que se desplace a la posición deseada.

La coordenada de la posición vertical de la carga (Eje Z) se mantiene invariable y se mide con un codificador rotatorio (*Encoder*), que es el sistema de medida instalado de fábrica en el propio puente grúa.

1.4. Estructura del documento

En la sección 2 se explica el contexto del Trabajo de Fin de Grado y se resumen las tesis en las que se ha basado.

En la sección 3 se explican brevemente todos los elementos físicos y programas con los que se ha trabajado, así como su utilidad en este proyecto, y se describe la estrategia de visión empleada para la consecución de las coordenadas de la carga en el plano horizontal.

En la sección 4 se describe la estructura y funcionamiento del puente grúa utilizado, así como las modificaciones realizadas para la introducción de las coordenadas de la carga en la realimentación del sistema de control.

En la sección 5 se explica el resultado del trabajo realizado y se compara con el sistema de control inicial.

Por último, en la sección 6 se describen las conclusiones obtenidas y se proponen posibles mejoras.

2. Contexto del Trabajo de Fin de Grado

Para la realización del presente Trabajo de Fin de Grado se utiliza un puente grúa didáctico de la marca INTECO [4] que incluye un tablero RT-DAC/USB2 I/O [5] para su control mediante ordenador. Este equipo lo proporciona el Departamento de Electrónica de la Escuela de Ingeniería y Arquitectura de la Universidad de Zaragoza, junto a un manual de instalación [6] y un manual de usuario [7].

Se parte de dos tesis que trabajan con el mismo puente grúa, y que se describen a continuación.

La tesis de control de un puente grúa con visión artificial [8], de la Universidad Tecnológica de Tallin, consiste en la creación de una aplicación que a través de una videocámara visualiza una marca en el suelo, dentro del espacio de trabajo del puente grúa, y calcula sus coordenadas con el objetivo de enviarlas a la entrada del sistema de control del puente grúa para que se sitúe encima de dicha marca. Todo ello utilizando el entorno Matlab – Simulink, del que se hablará posteriormente. Para ello coloca cuatro marcas circulares de color rojo en el suelo del puente grúa, una en cada esquina, y otra marca blanca dentro entre las cuatro marcas rojas y captura una instantánea cada vez que el usuario quiere. La instantánea se filtra para distinguir las marcas rojas del resto de la imagen y se calculan sus centros. Con las coordenadas de los centros, la imagen se ajusta en perspectiva obteniendo una imagen en planta sobre la que se detecta el centro de la marca blanca. El proceso se resume en la figura 2. Los principales inconvenientes que se encontraron, fueron la pérdida de precisión de los codificadores rotatorios del puente grúa tras la toma de unas pocas instantáneas, y el largo tiempo de espera desde la adquisición de la fotografía de la marca hasta la transmisión de ésta al puente grúa.

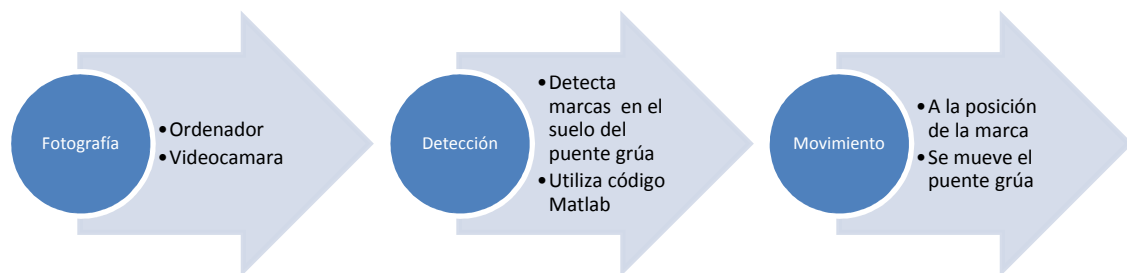


Figura 2. Diagrama de bloques de la tesis de control del puente grúa por visión artificial.

La tesis de modelado de un puente grúa [9], del Instituto de Electrónica de la Universidad Tecnológica de Tallin, consiste en la creación de un modelo matemático de representación de espacio de estados del puente grúa en el entorno Matlab-Simulink, con el fin de calcular las coordenadas de los tres ejes de la carga en todo momento y realimentar el sistema de control, sustituyendo de esta forma a los codificadores rotatorios instalados en el puente grúa y que realizaban la realimentación del sistema de control. En la figura 3 se muestra el esquema de esta tesis. La principal ventaja de este método es la eliminación del mantenimiento de los codificadores rotatorios, que pierden precisión a medida que se realizan movimientos consecutivos sin volver al origen del sistema de referencia del puente grúa y resetearlos. El inconveniente de este método es la pérdida de precisión respecto a la realidad, debido a que el modelo construido es ideal y se realizan aproximaciones en el comportamiento del puente grúa.

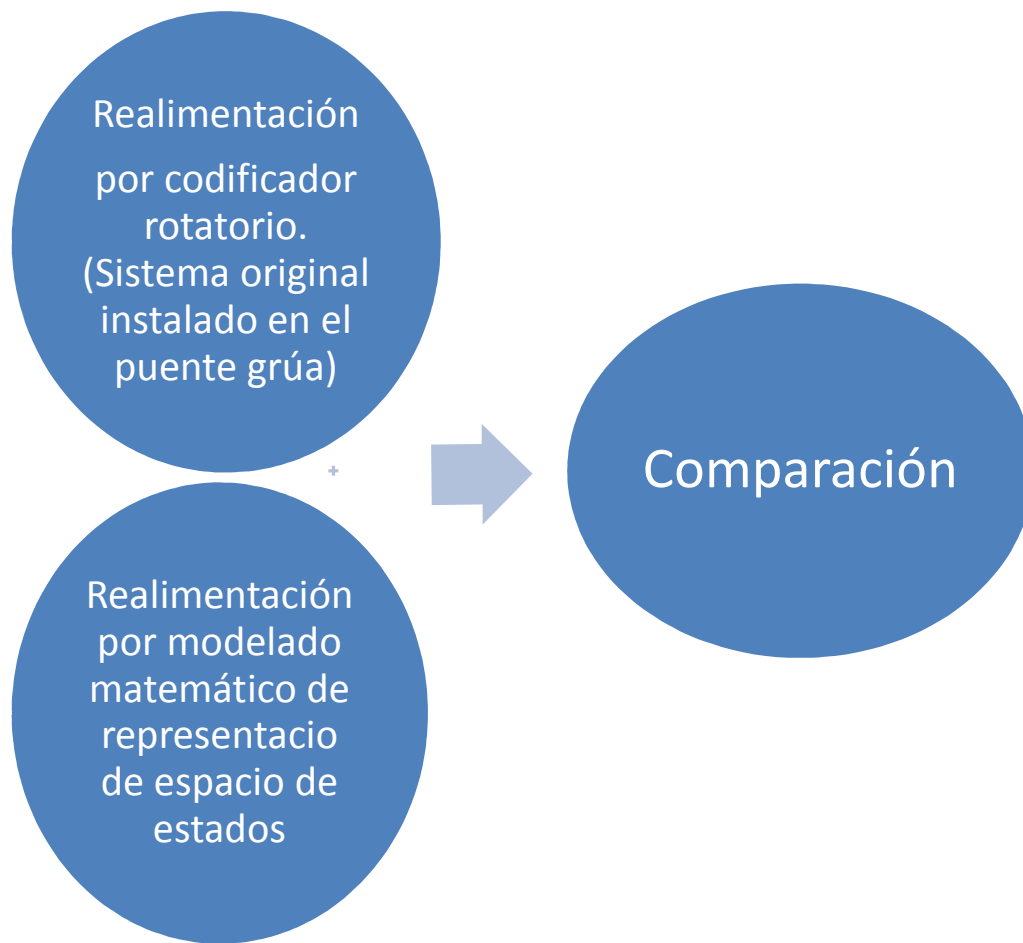


Figura 3. Esquema de la tesis de modelado y control de un puente grúa.

En el presente Trabajo de Fin de Grado, se quiere realizar un sistema de control mediante realimentación por videocámara de las coordenadas X e Y de la posición de la carga del puente grúa, con el objetivo de subsanar los inconvenientes de rapidez y precisión de las tesis mencionadas.

Para mejorar la rapidez de transmisión de las coordenadas de la videocámara al puente grúa, se decide grabar por videocámara en vez de tomar instantáneas de las marcas (las instantáneas se capturan del vídeo). Además, se decide utilizar el entorno Spyder - Anaconda con el lenguaje de programación Python en lugar del entorno Matlab para la obtención de las coordenadas, debido a que incluye una librería de visión artificial muy potente llamada Open CV. Estas coordenadas se envían a través de puertos de redes locales del ordenador, sin utilizar elementos externos. Al ordenador tan solo se conecta el puente grúa y la videocámara.

Además, se comparará la precisión de este sistema de realimentación al sistema de control del puente grúa respecto del sistema de realimentación por el codificador rotatorio instalado de fábrica, y respecto al sistema de realimentación mediante modelo matemático de representación de espacio de estados. En la figura 4 se muestra el esquema de este propósito.

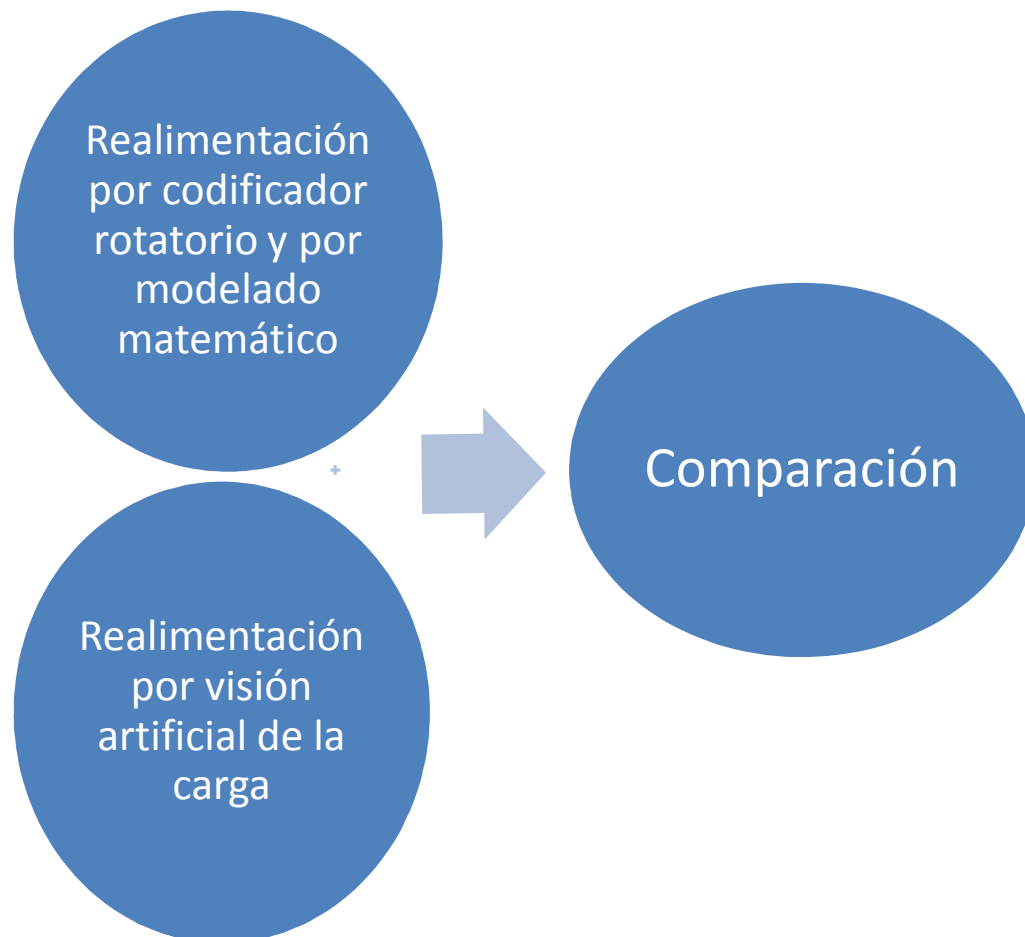


Figura 4. Esquema del objetivo del presente Trabajo de Fin de Grado.

De la primera tesis se coge la estrategia de colocación de cuatro marcas rojas circulares en el suelo para su reconocimiento por videocámara y se emplea el mismo ajuste de perspectiva para la obtención de una imagen en planta del espacio de trabajo del puente grúa. Debido a que el objetivo no es la introducción de las coordenadas de la posición de destino de la carga del puente grúa, sino la introducción de su posición en cada momento para la realimentación del sistema de control, se decide situar una quinta marca circular sobre la carga del puente grúa.

De la segunda tesis se cogen los resultados de realimentación por codificador rotatorio y por el modelo matemático de representación de espacio de estados creado por el autor de dicha tesis, para su posterior comparación.

3. Obtención de la posición de la carga por visión

En este apartado se describen los elementos físicos y programas utilizados para la obtención de la posición de la carga del puente grúa por visión. Además se describe la estrategia utilizada para su detección y obtención, así como todos los pasos que se han seguido.

3.2. Elementos físicos y programas utilizados

Para la realización de este Trabajo de Fin de Grado se han utilizado un puente grúa didáctico, un ordenador portátil, y una videocámara. En el ordenador además, se han instalado los software Matlab – Simulink y Spyder – Anaconda con el lenguaje Python y la librería Open CV. A continuación se detallan estos elementos.

3.2.1. Puente grúa didáctico de INTECO

Para conseguir el objetivo de este trabajo se ha utilizado un puente grúa didáctico proporcionado por INTECO. Este puente grúa es un sistema electromecánico no lineal de múltiples entradas y múltiples salidas que tiene un complejo comportamiento dinámico y presenta diversos desafíos para su control.

El hardware necesario para controlar el puente grúa, y cuya relación se muestra en la figura 5, es el siguiente:

- El propio puente grúa.
- Un ordenador con procesador Pentium o AMD. Para este proyecto se utiliza un ordenador portátil Asus K53SD.
- Un tablero RT-DAC/USB2 I/O, que es una placa multifunción de E/S (digital y analógica) dedicada a la adquisición de datos en tiempo real y que puede ser reconfigurada para nuevas funcionalidades de todas las entradas y salidas sin modificar el hardware. Contiene un chip FPGA de Xilinx, cuya configuración predeterminada acepta señales de codificadores rotatorios incrementales y genera salidas PWM (*Pulse Width Modulation*). Este elemento se puede observar en la figura 6.
- Una interfaz de potencia desde una fuente de alimentación a la grúa, en la que se distribuyen dos tipos de potencia. La primera con una conexión a una red de corriente alterna de 110/220 V, y la segunda con una conexión a una red corriente continua de 24 V.

El software necesario para controlar el puente grúa es el siguiente:

- Microsoft Windows XP/W7x86.
- MATLAB 32 bit con Simulink y la caja de herramientas RTW (Real Time Windows).
- Compilador Microsoft Visual C++.
- CD-ROM para la instalación de la caja de herramientas RT-CON para el control del puente grúa en tiempo real.
- Protocolo TCP/IP instalado.

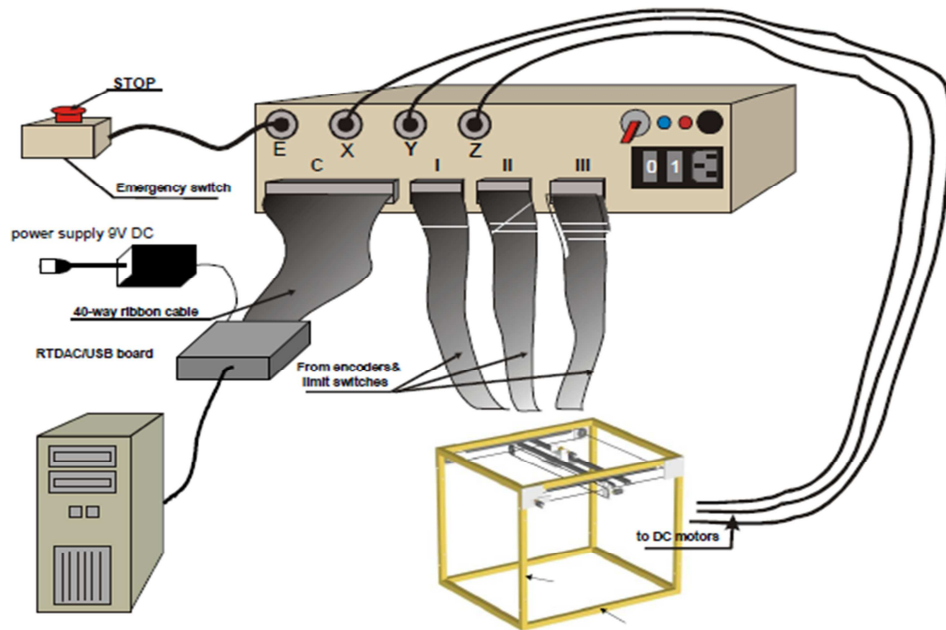


Figura 5. Esquema de la relación del hardware necesario para controlar el puente grúa.



Figura 6. Imagen del tablero RT-DAC/USB2.

3.2.2. Videocámara

El objeto principal de visión utilizado es una videocámara tipo *webcam* suministrada por el Departamento de Electrónica de la Escuela de Ingeniería. Tiene una resolución de 12 MP, adecuada para captar el movimiento de la carga del puente grúa en el espacio de trabajo. Se puede observar montada sobre un trípode en la figura 8.

3.2.3. Matlab

MATLAB (MATrix LABoratory) es un sistema de cómputo numérico que ofrece un entorno de desarrollo integrado con un lenguaje de programación propio y que permite analizar datos, desarrollar algoritmos y crear modelos y aplicaciones propias. Las aplicaciones de este sistema que son relevantes para el desarrollo del trabajo son: el control de sistemas, el procesamiento de imágenes y vídeo y el procesamiento de señales [10].

En este sistema se integra Simulink, una herramienta de programación adecuada para modelar, simular y analizar sistemas dinámicos. Esta herramienta se utiliza en ingeniería de control y procesamiento de señal para la simulación de sistemas y el diseño basado en modelos. Permite la introducción desde MATLAB de algoritmos dentro de los modelos y la exportación a MATLAB de los resultados de las simulaciones para su posterior análisis. Además, está altamente integrado con el resto de MATLAB.[11].

Además, se utiliza RTWT (*Real Time Windows Target*), una herramienta integrada en MATLAB que permite el prototipado y prueba de sistemas en tiempo real [12]. Para ello utiliza un pequeño *kernel* que se ejecuta en tiempo real utilizando el modo privilegiado de la CPU y el reloj incorporado en el ordenador. Para ello, dicho *kernel* intercepta la interrupción de la señal de reloj antes de que la reciba el sistema operativo, otorgando la máxima prioridad a la aplicación en tiempo real. Otra característica de esta herramienta es que da prioridad a las tareas rápidas sobre las tareas lentas. El *kernel* se comunica con el hardware de E/S mediante bloques de entrada/salida y con Simulink a través del módulo de interfaz del modo externo de Simulink. De este modo, permite ejecutar simulaciones en tiempo real en el puente grúa [13].

3.2.4. Open CV + Python + Spyder (Anaconda)

Python es un lenguaje de programación de alto nivel del tipo *scripting*, desarrollado como proyecto de código abierto y diseñado para ser sencillo de leer e implementar.

Permite la división de programas en módulos reutilizables desde otros programas Python y viene con una gran colección de módulos estándar que proporcionan E/S de ficheros, llamadas al sistema, etc [13].

Open CV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos, calibración de cámaras, visión estérea y visión robótica.

Cuenta con un desarrollo activo en interfaces para Python, Ruby, Matlab y otros lenguajes y ha sido diseñado para ser eficiente en cuanto a gasto de recursos computacionales y con un enfoque hacia las aplicaciones en tiempo real [14].

Anaconda es una distribución libre y abierta de los lenguajes Python y R, utilizada en ciencia de datos, y aprendizaje automático. Esto incluye procesamiento de grandes volúmenes de información, análisis predictivo y cómputos científicos. Está orientado a simplificar el despliegue y administración de los paquetes de software. Se utiliza como punto de partida para la instalación de Open CV [15].

Spyder es un potente entorno de desarrollo interactivo a través de Anaconda para el lenguaje Python. Posee funciones avanzadas de edición, pruebas interactivas, depuración e introspección y un entorno informático numérico [16].

3.3. Estrategia de visión

La estrategia de visión consiste en la colocación de unas marcas de referencia en el suelo donde se sitúa la estructura del puente grúa y de una marca objetivo en la carga del puente grúa, para su posterior seguimiento a través de la grabación de un vídeo con la videocámara y el posterior procesamiento de las capturas del mismo.

3.3.1. Dimensiones del espacio de trabajo del puente grúa

Las dimensiones del espacio de trabajo del puente grúa son 920 mm en el eje X (Eje del carro) y de 920 mm en el eje Y (Eje del carril), y de 820 mm en el eje Z (Eje del movimiento vertical de la carga).

En este trabajo se trabaja con una longitud de cuerda constante, por lo tanto el motor encargado de la variación de la posición en el eje Z no se utiliza.

Para leer la posición en el eje Z y los ángulos de desviación de la carga se utilizan los correspondientes codificadores rotatorios integrados en el propio puente grúa. Una mayor corrección de la desviación de la carga respecto al eje vertical del carro supone una menor oscilación de la misma, lo que permite una mayor precisión en el seguimiento de la carga por videocámara.

Para leer la posición en los ejes X e Y se utiliza una videocámara y se desarrolla una estrategia de visión junto a un programa que permite la detección de la carga y obtención de dichas coordenadas.

3.3.2. Colocación de las marcas

Se colocan cuatro marcas circulares en el suelo del espacio de trabajo de la grúa de tal forma que sus centros coincidan con las esquinas del plano del espacio de trabajo formado por los ejes X e Y , formando así un plano de coordenadas [(0,0), (0,920), (920,0), (920,920)] mm. Se decide que estas marcas sean de color rojo por su contraste con color del suelo donde se instala la grúa y con el color de los elementos de la misma. Tienen un diámetro de 90 mm y a partir de ahora se denominarán marcas de referencia.

Se coloca una quinta marca circular en la parte superior de la carga de la grúa de tal forma que el centro de la marca coincida con el centro de la carga. Esta marca es del mismo color que las marcas de referencia y tiene un diámetro de 40 mm, lo suficientemente inferior al de las marcas de referencia para su correcta distinción a la hora de procesar las imágenes. A partir de este momento, a esta marca situada en la carga del puente grúa se le denominará marca objetivo.

3.3.3. Colocación de la videocámara

Se coloca una videocámara sobre un trípode en la parte externa de un lateral de la estructura del puente grúa de forma que sea capaz de observar las cinco marcas en todo momento y ante cualquier movimiento de la carga. Con el fin de distinguir las marcas de referencia a la hora de obtener las coordenadas de sus centros, la cámara se orienta ligeramente hacia una esquina del espacio de trabajo para que en las imágenes procesadas ninguna marca tenga las mismas coordenadas ni en el eje X ni en el eje Y. Esta videocámara está conectada a un PC desde el que se pueden observar las imágenes en cualquier momento. En la figura 7 se puede observar el esquema de este proceso y en la figura 8 la colocación de las marcas y de la videocámara sobre el trípode y enfocando a las cinco marcas.

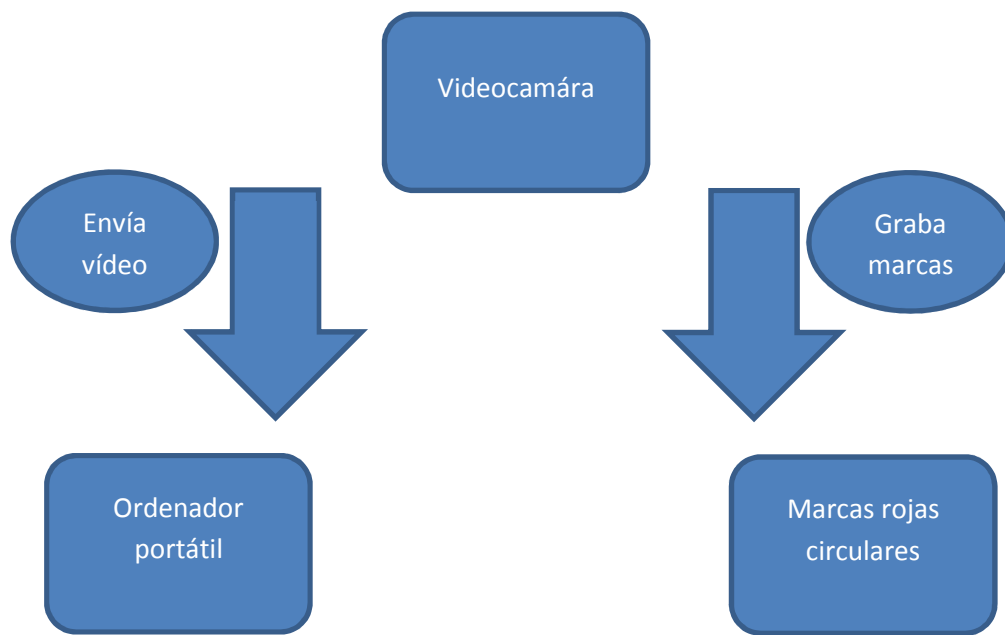


Figura 7. Esquema de la conexión de los elementos para la grabación de vídeo.

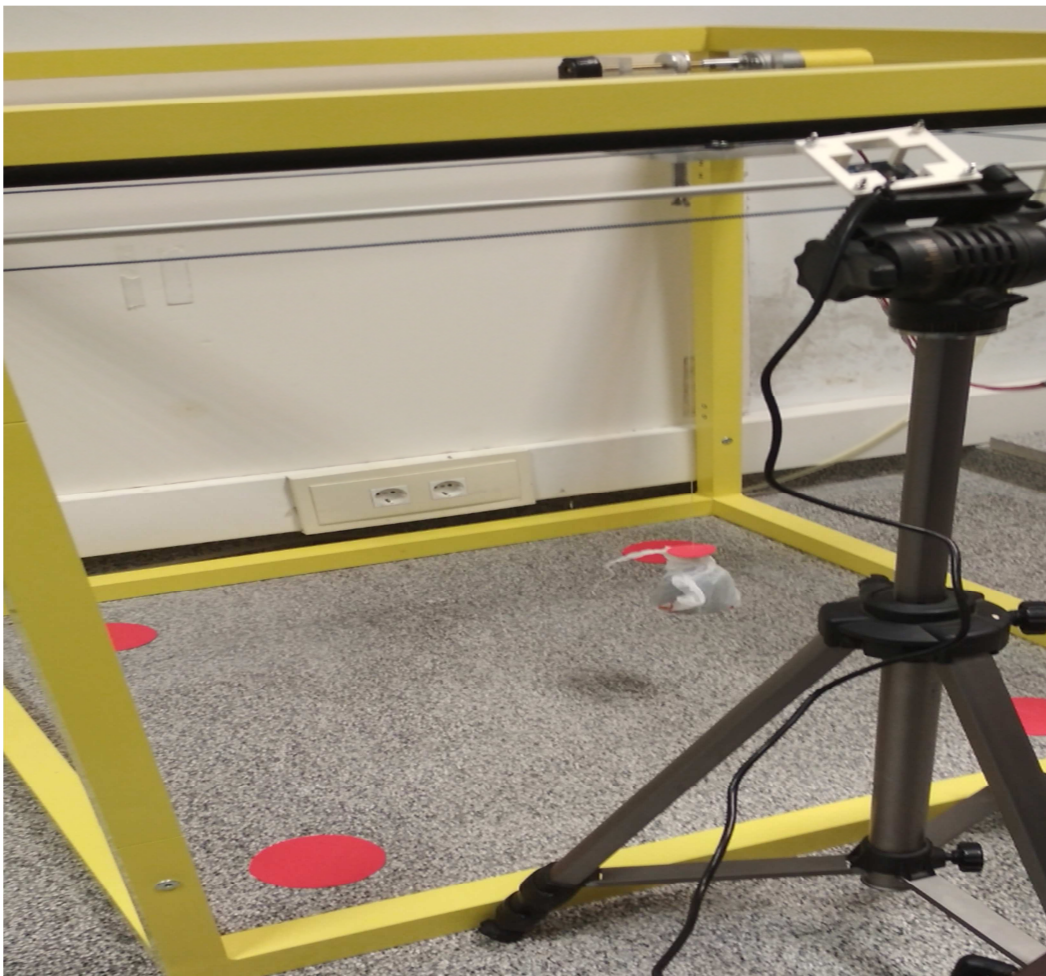


Figura 8. Colocación de las marcas y la videocámara.

3.4. Detección de las marcas por videocámara

En este apartado se describen las distintas fases para la grabación y filtrado de las capturas obtenidas en la grabación con el fin de distinguir las marcas del resto de la imagen. En el anexo 1 se pueden observar todos programas realizados para el ajuste de parámetros para conseguir las coordenadas de la carga del puente grúa, así como los resultados obtenidos en el ajuste de parámetros de dichos programas.

3.4.1. Inicio del programa y grabación de vídeo

En primer lugar, se importan las librerías numpy y cv2, la primera es una biblioteca de funciones matemáticas de alto nivel para operar con vectores o matrices y la segunda es la biblioteca para el tratamiento de imágenes Open CV. Tras esto, se enciende la cámara y se captura el video con el comando `cv2.VideoCapture(1)`, siendo 1 el número de la cámara webcam que se utiliza [17].

```
import numpy as np
import cv2
cap = cv2.VideoCapture(1)
```

Una vez iniciado el vídeo se capturan las imágenes del mismo cuadro a cuadro, con un bucle `while()` y el comando `cap.read()`. Sobre estas imágenes se trabajará posteriormente para reconocer las marcas de referencia y la marca objetivo. La imagen original capturada por la cámara se puede ver en la figura 9.

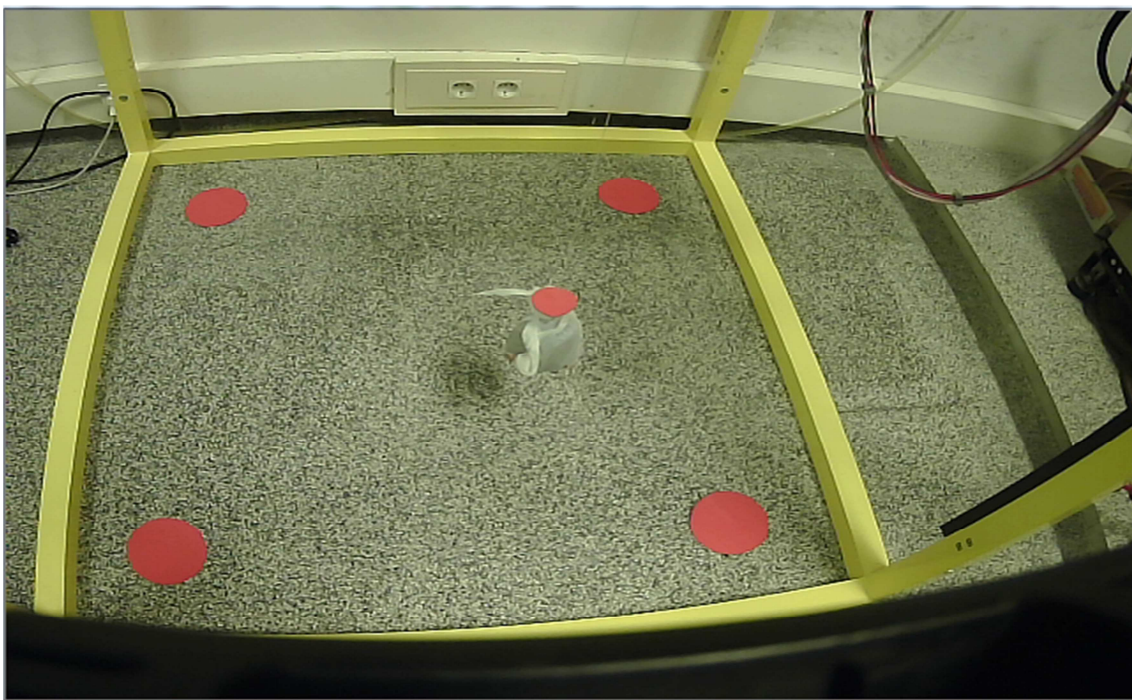


Figura 9. Imagen original captada por la cámara.

3.4.2. Obtención de la imagen binaria

Sobre las imágenes capturadas anteriormente se aplica un cambio de color a escala de grises utilizando la función: `gris = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)`, siendo `frame` la imagen original capturada y `gris` la imagen de salida en escala de grises.

A las imágenes en escala de grises se les aplica un filtro de mediana para suavizar la imagen. Para ello se crean núcleos de tamaño un número impar y se aplica el filtro sobre estos núcleos, de tal forma que el valor del píxel central de cada núcleo toma el valor medio del resto de píxeles del núcleo. Esto reduce el ruido que pueda haber en las capturas del vídeo. Se utiliza la función: `median = cv2.medianBlur(gris,k)`, siendo `k` el tamaño elegido para el núcleo y `median` la imagen filtrada que se obtiene.

Una vez realizados estos dos pasos, se procede a la umbralización, que es la conversión de la imagen en escala de grises y filtrada a escala binaria, donde cada píxel, en función de su valor en la escala de grises, toma un color blanco o negro, quedando una imagen formada por píxeles blancos y negros. Se utiliza la función: `ret,thresh1=cv2.threshold(median,1,255,cv2.THRESH_BINARY_INV)`, siendo `THRESH_BINARY_INV` el estilo de umbralización que se ha elegido, de tal forma que las marcas tengan color blanco y el resto de la imagen color negro; `1` el valor del umbral de los píxeles para la conversión a color blanco, `255` el valor que tendrán los píxeles que no superan el umbral, y `thresh1` la imagen binaria obtenida.

Se pueden observar en el anexo 1 las diversas pruebas con distintos valores del parámetro `1` para las imágenes obtenidas en escala de grises con cada parámetro `k`.

En la figura 10 se muestra la que se considera la mejor imagen obtenida con este método.

Se observa que hay demasiado ruido en la imagen como para ser eliminado con filtros de ruido sin eliminar ninguna de las marcas de interés, por lo que se prueba otro método en el cual se convierte la imagen original a HSV en lugar de a escala de grises y posteriormente se extraen los píxeles que están dentro del rango de los colores rojos, lo que significa que estos píxeles se tornan blancos y los del resto de la imagen, negros.

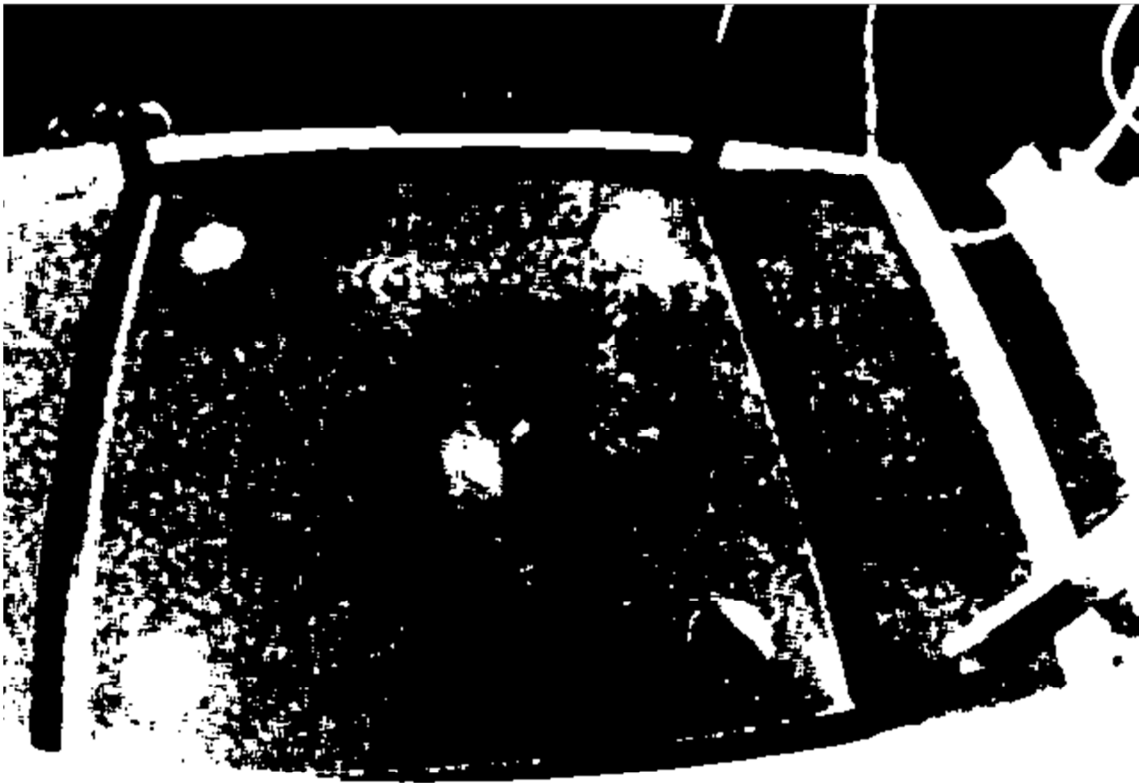


Figura 10. Imagen binaria por el método de umbralización.

El fragmento de programa utilizado es el siguiente:

```
hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
rojo_bajos1 = np.array([0,65,75], dtype=np.uint8)
rojo_altos1 = np.array([12, 255, 255], dtype=np.uint8)
rojo_bajos2 = np.array([240,65,75], dtype=np.uint8)
rojo_altos2 = np.array([256, 255, 255], dtype=np.uint8)
mascara_rojo1 = cv2.inRange(hsv, rojo_bajos1, rojo_altos1)
mascara_rojo2 = cv2.inRange(hsv, rojo_bajos2, rojo_altos2)
mask = cv2.add(mascara_rojo1, mascara_rojo2)
```

Como se puede ver, es necesario crear dos intervalos de valores en el rango de los rojos, ya que los rojos están al principio y al final de la rueda de color HSV. Tras crear estos dos intervalos, se unen y se extrae la imagen binaria con la función `cv2.add`. [18]. El resultado se puede ver en la figura 11.

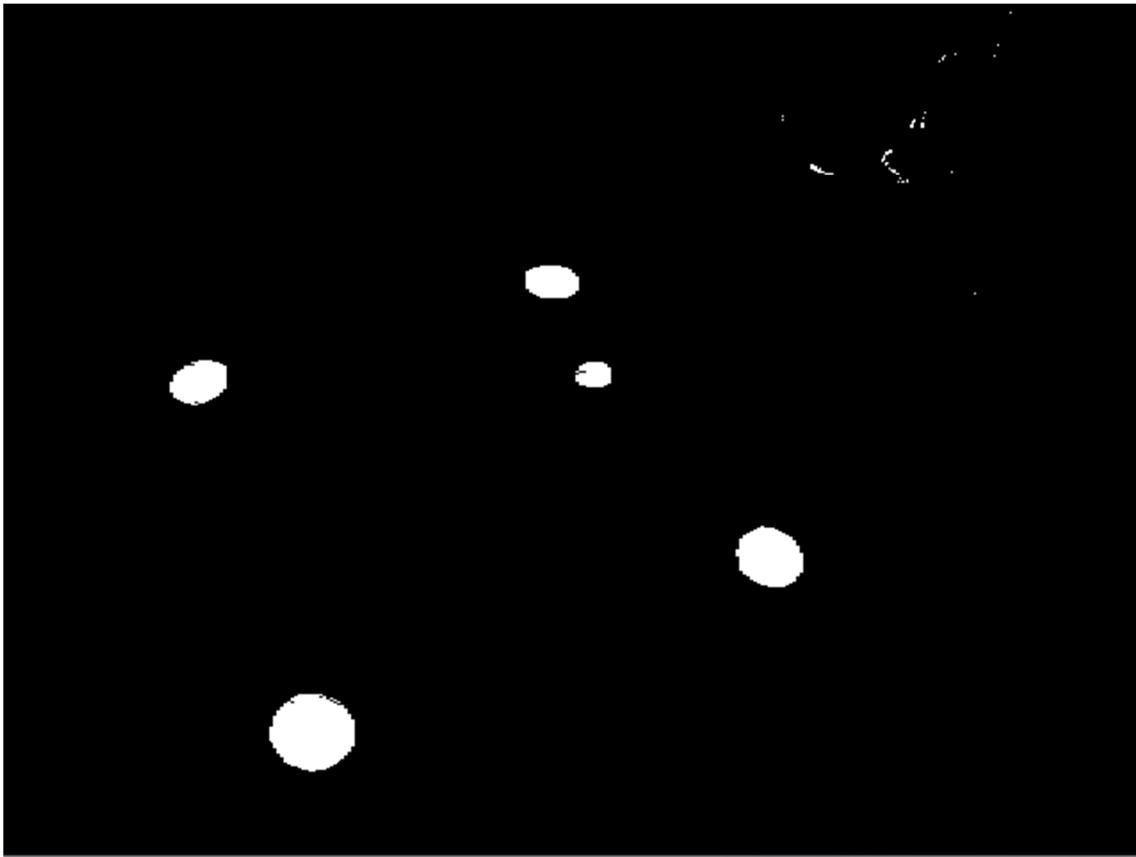


Figura 11. Imagen binaria obtenida por máscara de rojos.

3.4.3. Filtrado de ruido en la imagen binaria

Como se puede ver en la figura anterior, hay un poco de ruido en la imagen obtenida, producto de un cable rojo del propio puente. Para eliminarlo de la imagen se realiza un proceso de erosión-dilatación, adecuado para eliminar pequeños ruidos blancos. Para ello se crea un núcleo de tamaño (p,q) ; y se procede al proceso de erosión. Este proceso consiste en reducir de tamaño las marcas blancas, de tal forma que un píxel de la imagen original se considerará blanco si todos los píxeles dentro de la ventana del núcleo creado son blancos. Después de esto, se realiza el proceso de dilatación, que consiste en aumentar de nuevo el tamaño de las marcas de forma que un píxel de la imagen original será blanco si al menos uno de los píxeles dentro de la ventana de su núcleo lo es.

Este proceso se realiza con las siguientes funciones, siendo *dilatacion* la imagen de salida.

```
kernel = np.ones((p,q),np.uint8)
erosion = cv2.erode(thresh1,kernel,iterations = 1)
dilatacion = cv2.dilate(erosion,kernel,iterations = 1)
```

Se prueban distintos valores para los parámetros p y q , tal como se puede ver en el anexo, y se observa que con unos valores de 5,5 se elimina el ruido mencionado anteriormente. Se establecen unos valores de 7,7 por si por si este ruido aumentara debido al movimiento del puente grúa. La figura 12 muestra los resultados obtenidos con los parámetros establecidos.

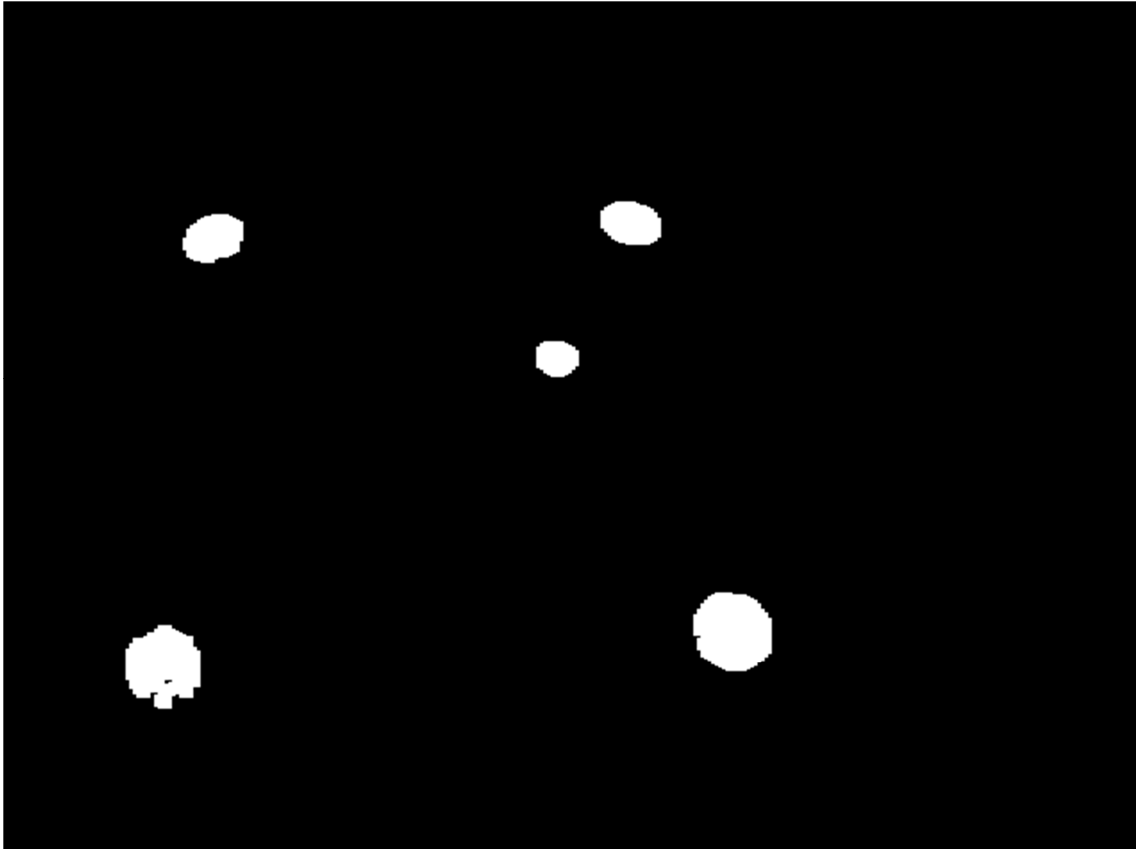


Figura 12. Imagen tras filtrado de ruido.

3.5. Obtención de la posición de la carga

En este apartado se describen los pasos a realizar para obtener las coordenadas de la posición del centro de la carga del puente grúa.

3.5.1. Obtención de los centros de las marcas de referencia

Tras obtener la imagen dilatación, se procede a la distinción de las marcas de referencia de la marca objetivo. Este proceso se realiza calculando los centros de todas las marcas cuya área este dentro de un intervalo de tal forma que solo se calculen los centros de las marcas de referencia. El área de las marcas de referencia es notablemente mayor a la de la marca objetivo. Para ello se buscan los contornos de las marcas, que es donde cambia el gradiente de la imagen binaria, y se calculan los momentos de la forma de las marcas y una vez obtenidos se calculan los centros.

Con el fin de conocer las coordenadas de los centros y el área de las cinco marcas, para después poder distinguir las marcas de referencia de la marca objetivo y verificar que las coordenadas obtenidas son correctas, se realiza un programa que nos indica el valor de las coordenadas de las marcas en el sistema de referencia de la imagen, tal y como se muestra en la figura 13.

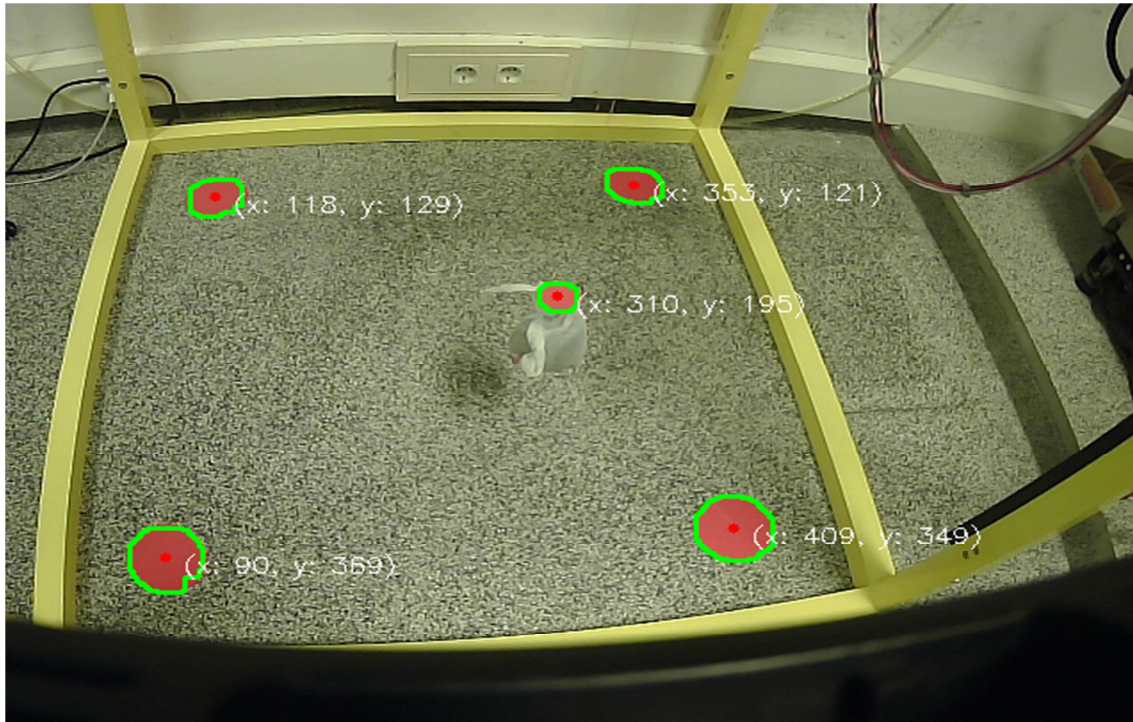


Figura 13. Imagen de los centros de las marcas.

Se observa que el origen del sistema de coordenadas está en la esquina superior izquierda y que el área de la marca objetivo es menor de 400, mientras que el área de todas las marcas de referencia es mayor que dicho valor. Por lo tanto, 400 será el valor elegido para diferenciar las marcas de referencia de la marca objetivo.

El fragmento del programa que permite calcular los centros de las cuatro marcas de referencia es el siguiente:

```

contours,_ = cv2.findContours(dilatacion, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)
cv2.drawContours(frame, contours, -1, (0,255,0), 2)
a=0
x=[0,0,0,0]
y=[0,0,0,0]
for i in contours:
    momentos = cv2.moments(i)
    area = cv2.contourArea(i)
    if area > 400:
        cx = int(momentos['m10']/momentos['m00'])
        cy = int(momentos['m01']/momentos['m00'])
        x[a] = cx
        y[a] = cy
        a=a+1
        cv2.circle(frame, (cx, cy), 3, (0,0,255), -1)
        cv2.putText(frame, "x: " + str(cx) + ", y: " + str(cy) +
        ")", (cx+10,cy+10), font, 0.5, (255,255,255), 1)
print (x)
print (y)
x2=sorted(x)
y2=sorted(y)
print(x1)
print(y1)

```

Las coordenadas obtenidas de los cuatro centros de las marcas de referencia se almacenan en las cadenas x, e y, siendo (x[0],y[0]) las coordenadas del centro de la primera marca detectada, (x[1],y[1]) las coordenadas del centro de la segunda marca, (x[2],y[2]) las coordenadas del centro de la tercera marca y (x[3],y[3]) las coordenadas del centro de la cuarta y última marca detectada.

Una vez obtenidas las coordenadas de los centros de las cuatro marcas de referencia, al no conocer el orden con el que se obtienen, se ordenan las cadenas x e y de menor a mayor. Las cadenas ordenadas se obtienen con la función `sorted`. Después de ordenar las coordenadas anteriores, se pueden relacionar las marcas obtenidas en la figura con sus coordenadas:

- El centro de la marca superior izquierda es ((x2(0), y2(1))
- El centro de la marca superior derecha es (x2(2), y2(0))
- El centro de la marca inferior derecha es (x2(3), y2(2))
- El centro de la marca inferior izquierda es (x2(1), y2(3))

Los centros de las marcas obtenidas se pueden observar en la figura 14.

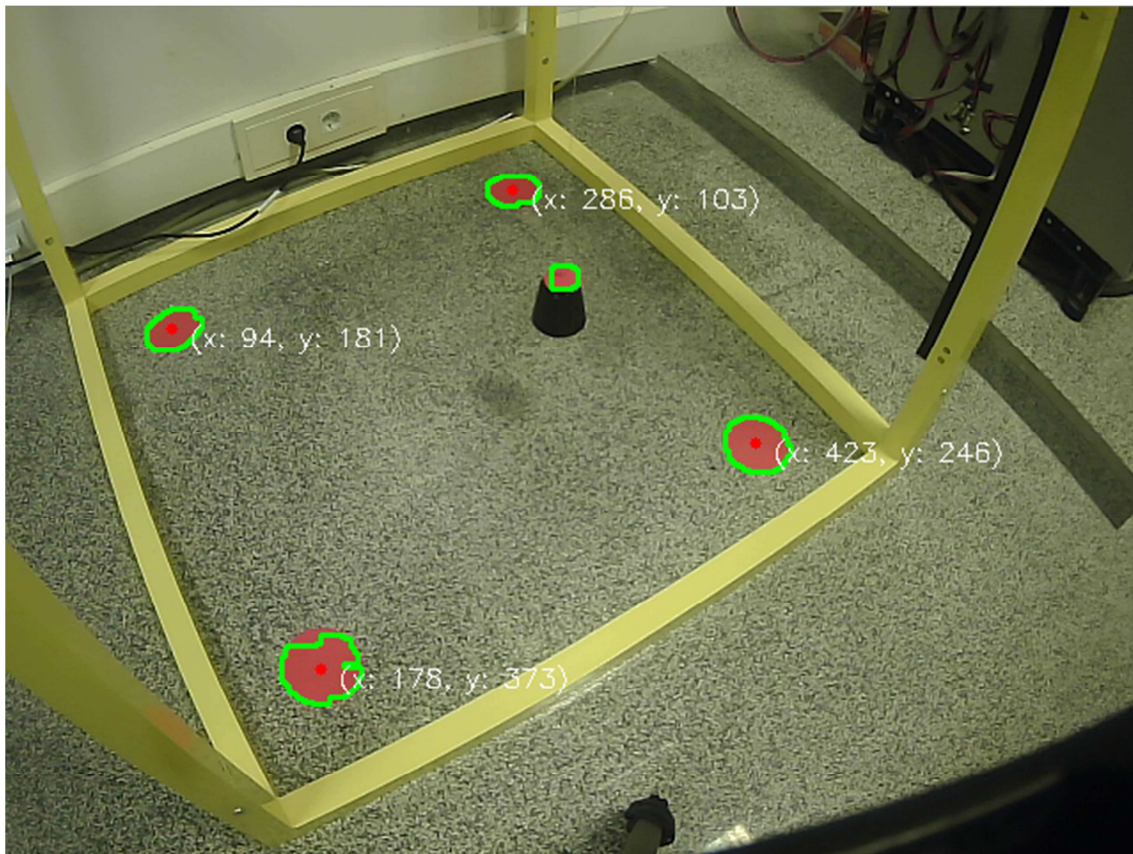


Figura 14. Imagen de los centros de las marcas de referencia.

3.5.2. Transformación geométrica de la imagen binaria

Ahora se realiza una transformación geométrica de perspectiva para obtener una imagen en planta a partir de la imagen observada por la videocámara. Esta imagen obtenida es rectangular y tiene por esquinas los centros de las marcas de referencia.

Este proceso se hace creando una matriz 3x3 con la función `cv2.getPerspectiveTransform()` que tiene como entrada las coordenadas obtenidas por videocámara de los centros de las marcas, y las coordenadas reales de los centros de las marcas en el plano del espacio de trabajo. Para realizar la transformación se utiliza la función `cv2.warpPerspective()`, cuyas entradas son la imagen que se quiere transformar, la matriz de transformación y el tamaño de la imagen obtenida, y cuya salida es la imagen en planta `imgplanta`. Para realizar correctamente dicha transformación es necesario conocer que el origen de coordenadas de la imagen está en la esquina superior izquierda, que el eje x es horizontal y el eje y vertical.

El fragmento de programa es el siguiente:

```

rows,cols,ch = frame.shape
pts1 = np.float32([[x2[0],y2[1]], [x2[2],y2[0]], [x2[1],y2[3]],
[x2[3],y2[2]]])
pts2 = np.float32([[0,0],[920,0],[0,920],[920,920]])
M = cv2.getPerspectiveTransform(pts1,pts2)
imgplanta = cv2.warpPerspective(dilatacion,M,(920,920))

```

Con el programa de imagen en planta, que se puede ver en el anexo 1, se obtienen los resultados que se muestran en la figura 15.

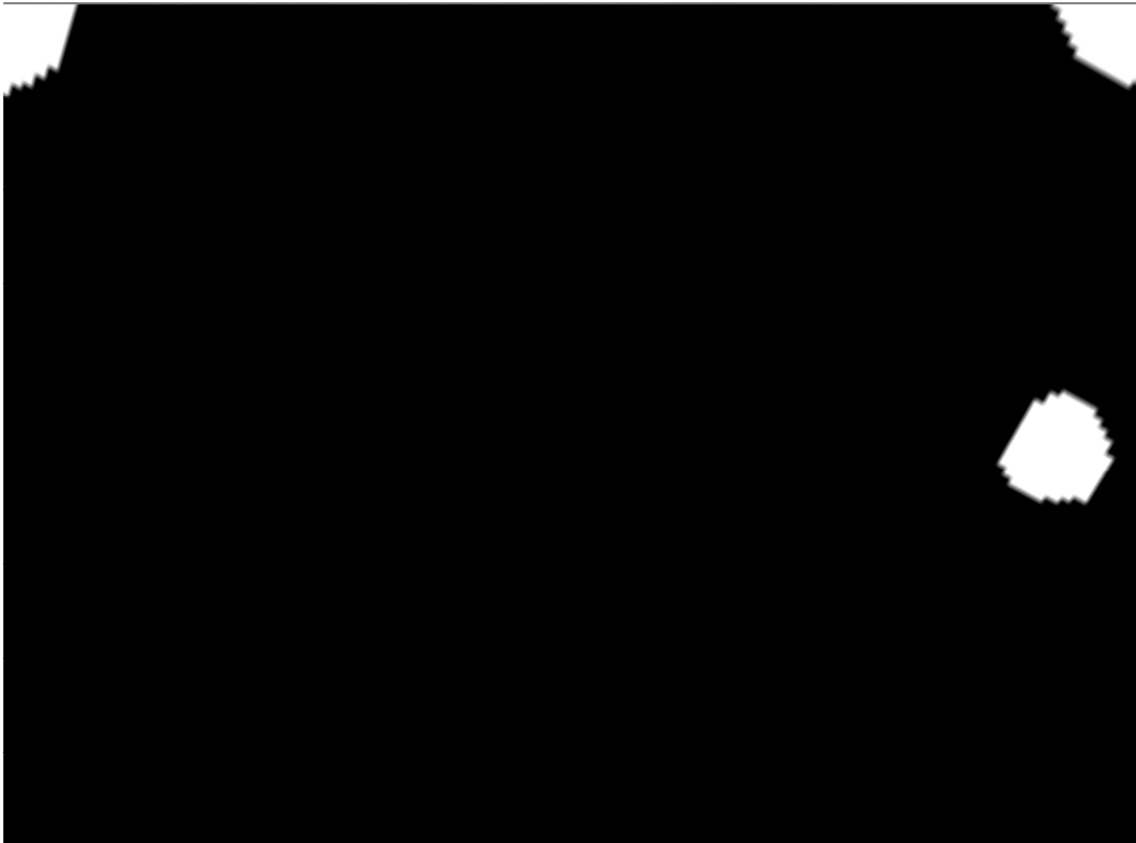


Figura 15. Imagen en planta obtenida.

3.5.3. Obtención del centro de la marca objetivo

Una vez que se obtiene la imagen en planta, y debido a que el origen del sistema de referencia del puente grúa se encuentra en la esquina opuesta del origen del sistema de referencia de la imagen obtenida por videocámara, se gira la imagen en planta 180 grados. Tras esto, se busca el contorno de la marca objetivo en la imagen rotada y se obtienen las coordenadas de su centro. Con este fin se crea un programa que imprime el área de los contornos detectados y se observa que el área de la marca objetivo es mayor que las áreas de los cuartos de círculo de las marcas de referencia que quedan en la imagen en planta, y se establece que con esta posición de la videocámara, un

umbral de área igual a 5000 es adecuado para distinguir la marca objetivo sobre el resto de contornos.

El fragmento de programa es el siguiente:

```
contours2, _ = cv2.findContours(imgplanta, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)
for i in contours2:
    momentos = cv2.moments(i)
    area = cv2.contourArea(i)
    if area > 5000:
        ox = int(momentos['m10']/momentos['m00'])
        oy = int(momentos['m01']/momentos['m00'])
```

En la figura 16 se observan las coordenadas obtenidas en los ejes X e Y de la marca objetivo, como resultado de dicha aplicación cuando la carga está en la situación mostrada en la figura 15, y sin rotar la imagen.

849
360
849
360
850
359
850
359
850
359

Figura 16. Captura de pantalla del centro de la marca objetivo

4. El Puente grúa de INTECO

En esta sección se describen la estructura mecánica del puente grúa y su comportamiento, así como el sistema de control del mismo. Además, se explican las simplificaciones realizadas sobre el comportamiento del puente grúa y las modificaciones efectuadas en su sistema de control. Las características del puente grúa, así como las de los motores y de los *encoder* que lleva acoplados, se muestran en el anexo 2.

4.2. Estructura del puente grúa

El puente grúa consiste en una estructura de forma cúbica que está formada por barras huecas unidas entre sí mediante conectores. Las dimensiones de longitud, anchura y altura de la estructura son **1000** mm para las tres magnitudes.

Sobre la parte superior de la estructura se monta el puente grúa, cuyos componentes principales son un carril, un carro y una carga. El carro se monta sobre el carril y la carga cuelga de un hilo perteneciente al sistema de elevación situado debajo del carro.

Para el movimiento de los tres componentes anteriores se utilizan tres motores de corriente continua RH158.24.75 DC fabricados por la empresa Micro Motors S.R.L.

Para medir los movimientos de la carga se utilizan cinco *encoder* con una resolución de 4096 pulsos por rotación fabricados por la misma empresa que los motores.

En la figura 17 se pueden observar los componentes del puente grúa.

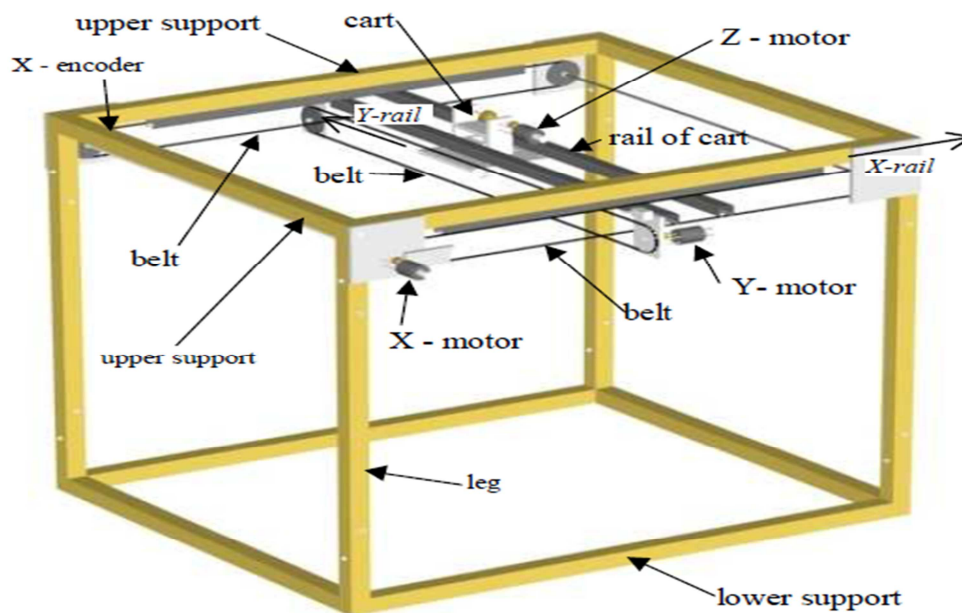


Figura 17. Esquema de los componentes del puente grúa.

El puente grúa se controla desde un PC con Matlab/Simulink.

Para comunicarse con el puente grúa desde el PC se utiliza un tablero RT-DAC/PCI que tiene integrado un chip programable fabricado por Xilinx. Este chip tiene la lógica necesaria para leer las medidas de los *encoder* y enviar señales a los motores. Tanto estas como el resto de las funciones del chip son accesibles y editables desde el PC a través de la caja de herramientas 3D Crane instalada en Matlab/Simulink. En la figura 18 se muestra la relación entre los distintos elementos mencionados.

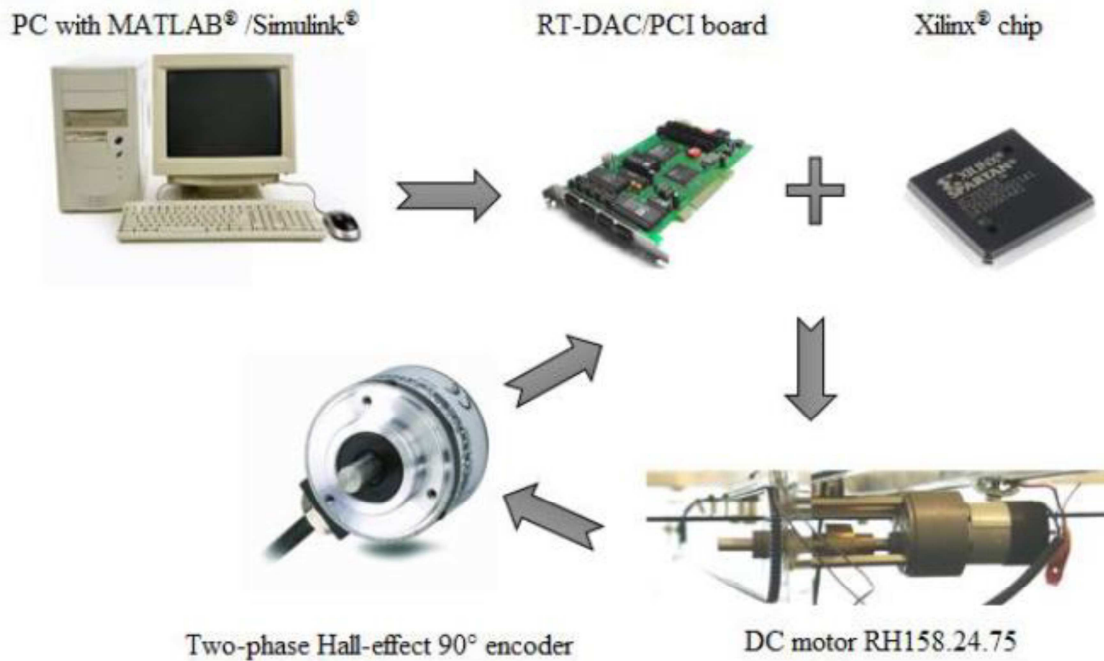


Figura 18. Relación entre los componentes del puente grúa.

4.3. Comportamiento del puente grúa

El comportamiento del puente grúa es el siguiente: El motor asociado al carril es capaz de mover este en la dirección que se denominará eje X. El *encoder* asociado a este eje mide el desplazamiento del carril en el eje X. El motor asociado al carro es capaz de mover el carro en la dirección longitudinal del carril, que se denominará eje Y. El *encoder* asociado a este eje mide el desplazamiento del carro en el eje Y. En el carro hay un sistema de elevación accionado por el motor restante que permite el ascenso/descenso de la carga en la dirección vertical, que se denominará eje Z. El *encoder* asociado a este sistema de elevación mide el desplazamiento de la carga en el eje Z.

Debido a estos movimientos se produce una oscilación de la carga, lo que se traduce en desviaciones de la carga respecto de la vertical del carro. Para medir dicha oscilación se utilizan dos *encoder*, uno que mide la desviación vista desde el eje Y

respecto del eje Z y que se denomina ángulo α , y otro que mide la desviación respecto del eje Z vista desde el eje X. A este último ángulo se le denomina ángulo β . Los *encoder* y los motores del puente grúa se pueden observar en la figura 19.



Figura 19. Derecha: Motor DC del puente grúa. Izquierda: *Encoder* del puente grúa.

4.4. Sistema de control del puente grúa

El puente grúa se controla a través de la caja de herramientas 3D Crane instalada en el entorno Matlab/Simulink en el PC que se utiliza. Se llama a la función Cr desde Matlab para acceder a la ventana principal de la interfaz gráfica del puente grúa que se divide en cinco partes principales: Herramientas de testeo, del puente grúa, controladores, y modelos y aplicaciones demo tanto con el puente grúa real como en simulación, tal y como se observa en la figura 20. Los esquemas de los sistemas de control demo se muestran en el anexo 3.

Desde esta ventana principal se pueden modificar los sistemas de control así como los parámetros del modelo del puente grúa y de los controladores. El objetivo de estas modificaciones es obtener un sistema robusto de control que envíe señales PWM a los motores en función de las señales de las coordenadas de posición y de los ángulos de desviación a través de los *encoder*.

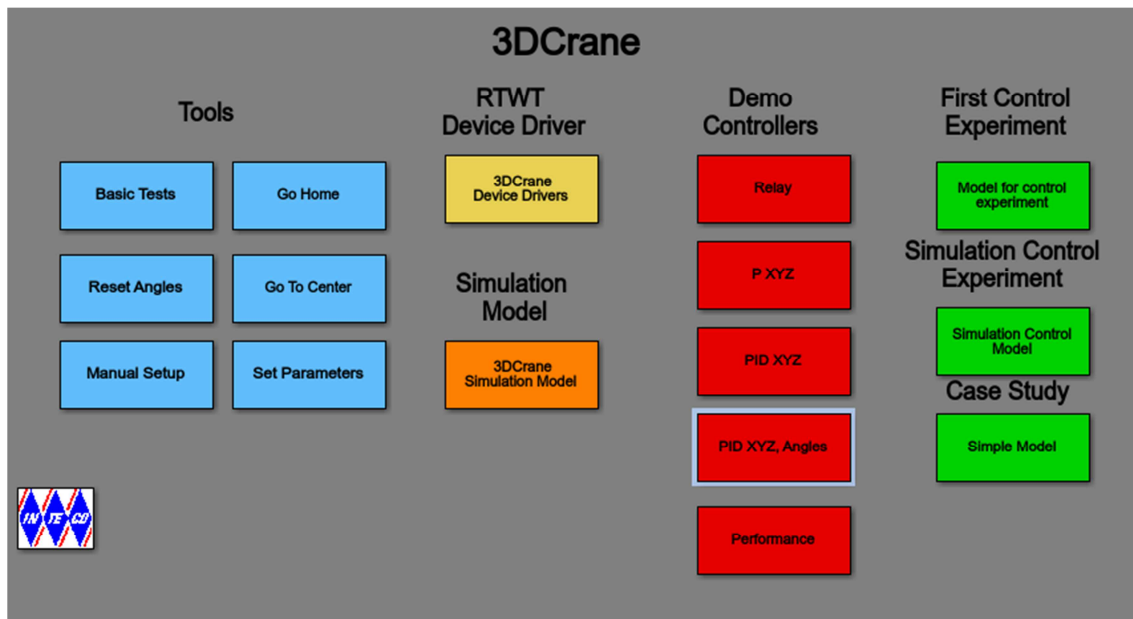


Figura 20. Captura de pantalla de la interfaz gráfica del puente grúa.

Para el desarrollo del proyecto presente se han realizado las siguientes modificaciones en el funcionamiento del puente grúa:

La longitud de la cuerda es constante, por lo tanto el motor asociado al sistema de elevación de la carga permanecerá inutilizado.

La longitud de la cuerda, así como los ángulos de desviación se leen desde los *encoder* correspondientes, mientras que la posición de la carga en el eje X y en el eje Y se lee desde el programa de visión anterior mediante una comunicación UDP.

4.4.1. Sistemas de control proporcionados

Tras analizar todos los sistemas de control demo incluidos en la caja de herramientas 3D Crane, se elige el sistema de control PID XYZ, Angles. Este sistema de control contiene cinco controladores PID, tres para el control en los ejes X, Y, Z; y dos para el control de los ángulos α y β , como se observa en la figura 21.

Aunque el objetivo de este trabajo no se centra en el control de los ángulos, la introducción de controladores PID a estos se traduce en una reducción de la oscilación de la carga, lo que permite una mayor precisión en el control de la posición en los ejes X e Y en el control por visión.

Además incluye tres bloques de saturación, uno por cada eje de posición, para limitar las señales PWM de entrada al modelo; y tres bloques de ganancias, también uno por cada eje de posición, que permiten el análisis de cada eje por separado.

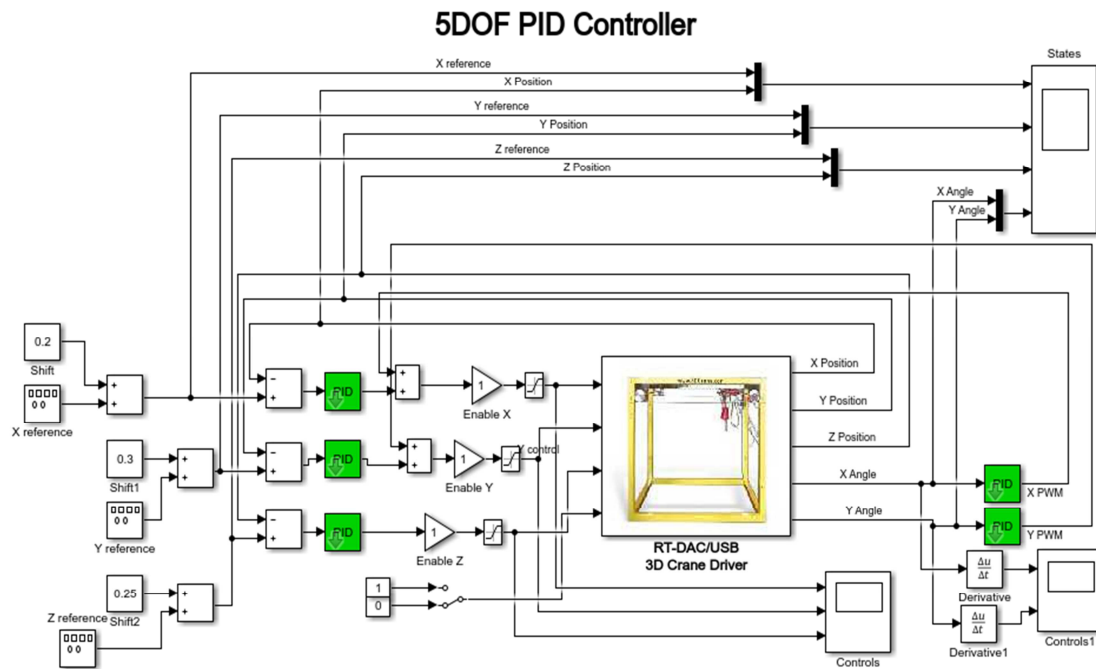


Figura 21. Captura del sistema de control elegido.

Se observa que se este sistema de control contiene tres osciloscopios: El primero con las señales de referencia y de salida de posición en los ejes X, Y, Z, y con las señales de salida de los ángulos α y β ; El segundo con las señales de control en los ejes X, Y, Z; y el tercero con las señales de control de los ángulos α y β .

4.4.2. Ajuste de los parámetros de los controladores

Se modifica este sistema para obtener en un osciloscopio las señales de referencia, de salida y de control del eje X, y del ángulo desviación α ; y en otro osciloscopio las señales de referencia, de salida y de control del eje Y, y del ángulo desviación β . El sistema modificado se puede ver en la figura 22.

El sistema de control anterior se analiza con el modelo de simulación proporcionado en la caja de herramientas 3D Crane con el fin de obtener unos parámetros adecuados en los controladores PID para analizar posteriormente el comportamiento del puente grúa real con dichos parámetros.

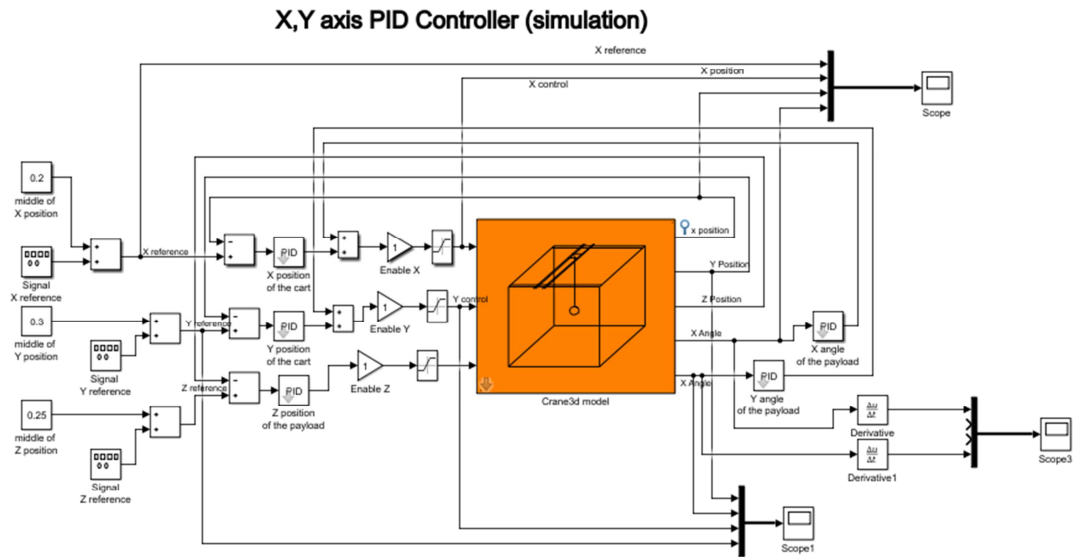


Figura 22. Sistema de control modificado para simulación.

Este modelo de simulación se construyó modelando matemáticamente el comportamiento del puente grúa y su comportamiento es ligeramente distinto al comportamiento del puente grúa real, por lo tanto es posible que haya que realizar una pequeña modificación en los parámetros de los controladores a la hora de controlar el sistema real.

La ejecución del modelo con los parámetros de los controladores PID proporcionados sin modificar proporciona los resultados mostrados en la figura 23 para el eje X y en la figura 24 para el eje Y. Se puede observar una sobreoscilación de la posición en ambos ejes que se considera excesiva, por lo que se decide ajustar los parámetros de los controladores. Algunas pruebas de ajuste de los parámetros de los controladores, así como los resultados obtenidos con dichos parámetros, se pueden observar en el anexo 4.

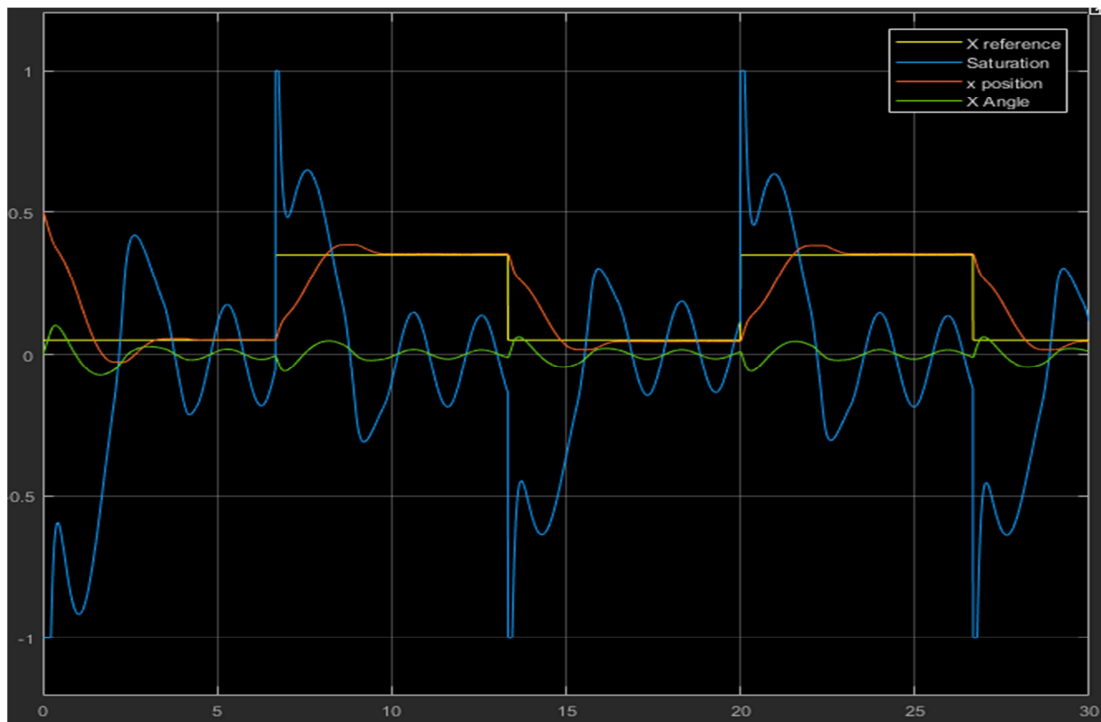


Figura 23. Señales de posición, oscilación y control en el eje X.

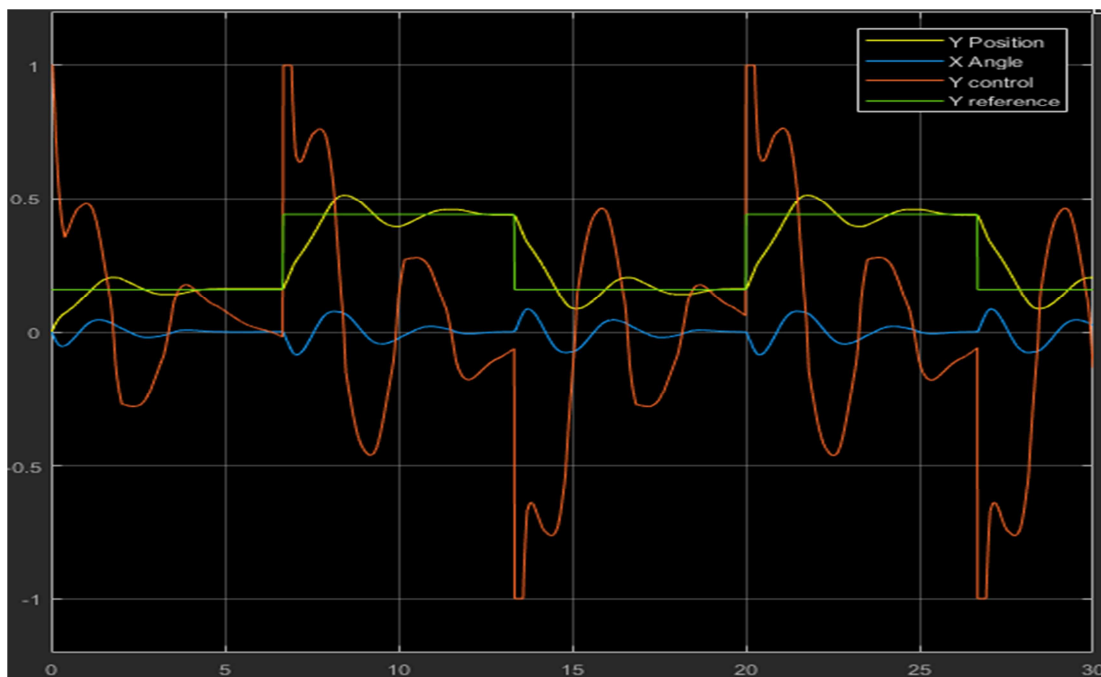


Figura 24. Señales de posición, oscilación y control en el eje Y.

Se modifican los parámetros de los controladores PID de los ejes X e Y, y de los ángulos α y β , y se observa que al aumentar los valores de la parte proporcional de los controladores de los ejes respecto de la parte proporcional de los controladores de los ángulos, se produce una respuesta rápida en posición, pero también una oscilación de magnitud indeseable de la carga. Por el contrario, al aumentar los valores de la parte proporcional de los controladores de los ángulos respecto de los valores de la parte proporcional de los controladores de los ejes, se observa una oscilación mínima de la carga, pero una respuesta lenta en posición e incluso no llega a alcanzar la posición deseada si la diferencia entre dichos valores es elevada.

Al aumentar el valor de la parte integral, se produce una sobreoscilación de la carga, lo que significa para nuestro sistema una pérdida de precisión.

Tras un proceso de prueba y error, cuyos resultados se muestran en el anexo, se establecen los valores que se muestran en la tabla. Y se obtienen los resultados para los ejes X e Y que se muestran en la figura 19 y en la figura 20, respectivamente. Se puede ver que ha disminuido la sobreoscilación en la posición a cambio de no llegar a la posición exacta. Este cambio se ha realizado para evitar choques del carro y del carril contra los topes del puente grúa a cambio de sacrificar la rapidez del movimiento, ya que la velocidad no es objeto de estudio en este Trabajo de Fin de Grado. Se observa también que la carga no oscila, cuando llega a la posición, igual que con los ajustes originales.

	Parte proporcional	Parte derivativa	Parte integral
Eje X	3	0	0
Eje Y	3	0	0
Ángulo α	5	0	0
Ángulo β	5	0	0

Tabla 1, Parámetros elegidos para los controladores de nuestro sistema de control.

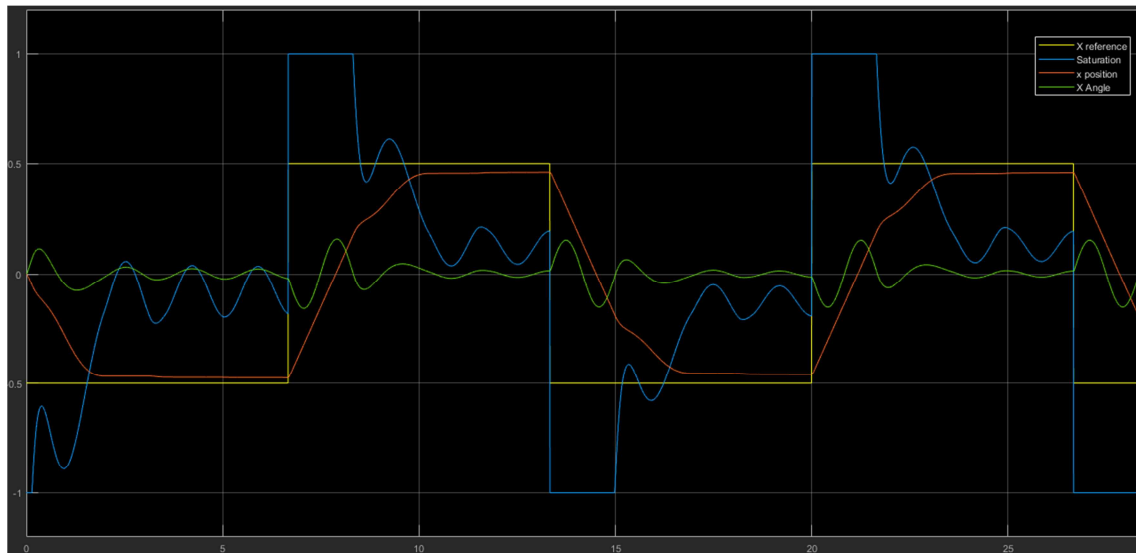


Figura 25. Señales de posición, oscilación y control en el eje X con PID ajustado.

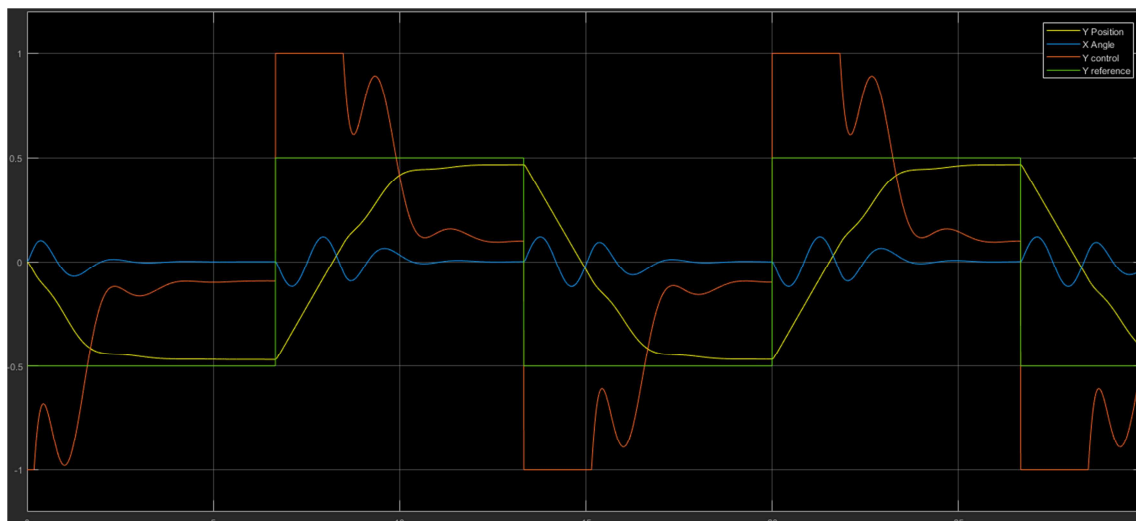


Figura 26. Señales de posición, oscilación y control en el eje Y con PID ajustado.

4.5. Comunicación Spyder – Simulink

Uno de los problemas afrontados en el trabajo ha sido el de la transmisión de las coordenadas de la carga desde Spyder a Matlab utilizando únicamente un ordenador. Para ello se probó con una aplicación de la compañía de desarrollo de software Eltima Software [19]. Aunque se comprobó una correcta transmisión de datos, tal y como se muestra en el anexo 5, esta opción se descartó debido a que el plazo de prueba gratuita que ofrecía la compañía de dicho software se consideró insuficiente.

Tras el anterior inconveniente, se decide probar una comunicación entre procesos mediante sockets a través de una red local lógica del ordenador. Se envían con el

protocolo de datagramas de usuario (UDP), que permite una transmisión en redes basadas en IP (Protocolo de internet) sin necesidad de establecer una conexión previa entre emisor y receptor, consiguiendo de esta forma una transmisión más rápida [20].

Debido a que con el protocolo de datagramas de usuario se pueden perder datos, se decide comprobar una correcta y precisa transmisión de datos. Con tal fin, se crea un programa en Spyder que suma números reales aleatorios comprendidos en el intervalo (0,1) hasta llegar a 30, y los envía cada 0.5 segundos utilizando la comunicación citada. Este programa se puede ver en el anexo 5. Estos números se reciben en Simulink a través del bloque 'UDP receive' [21] y se representan en un osciloscopio. Esta representación se puede observar en la figura 27.

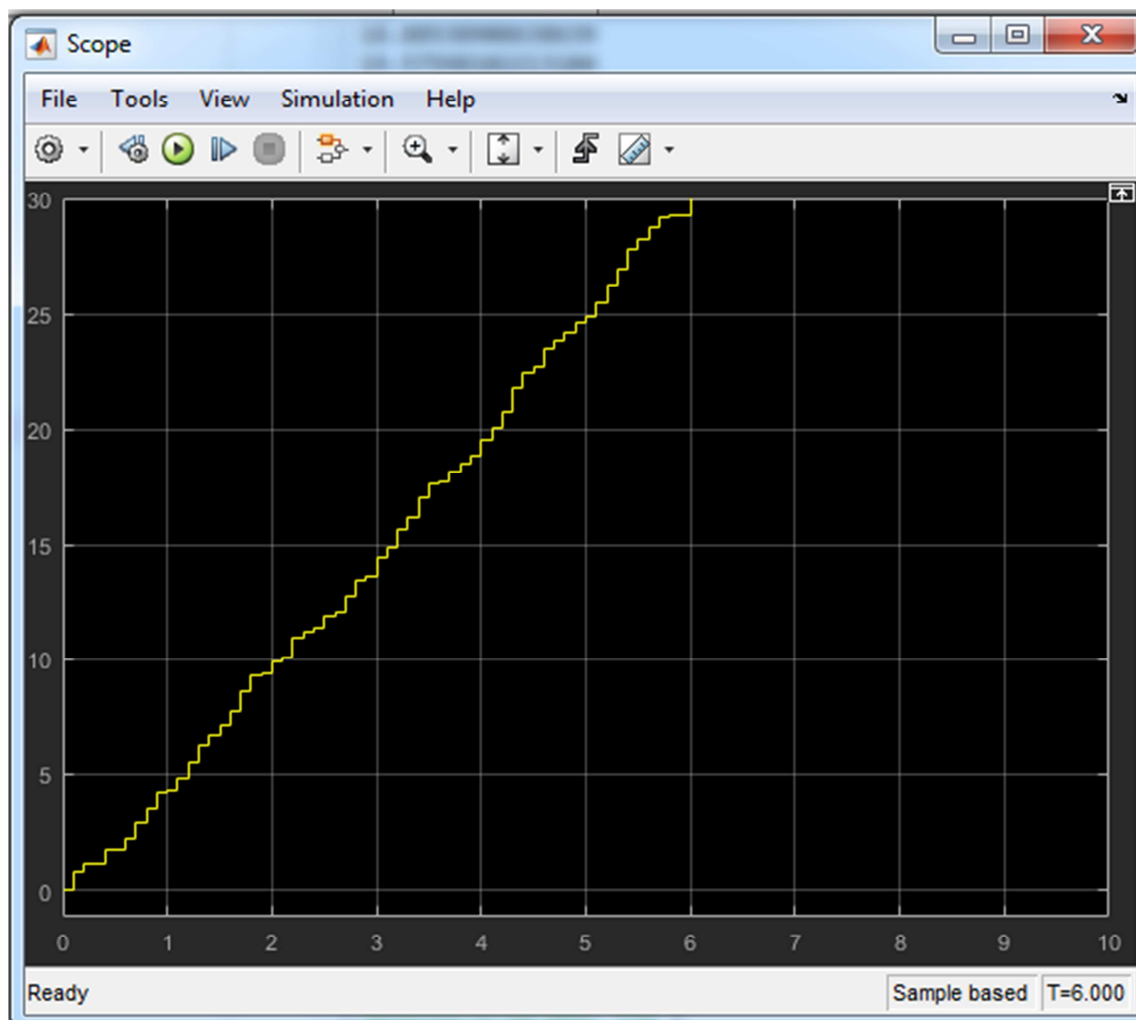


Figura 27. Representación de los datos recibidos a lo largo del tiempo.

Además, desde Matlab se comprueba la coincidencia de los decimales de los números enviados y recibidos, demostrando así una gran precisión en la transmisión de las

coordenadas utilizando esta comunicación (hasta seis decimales), tal y como se puede observar en las figuras 28 y 29. Por lo tanto se decide que este será el método de comunicación entre Spyder y Simulink. El programa completo de Spyder se muestra en el anexo 6.

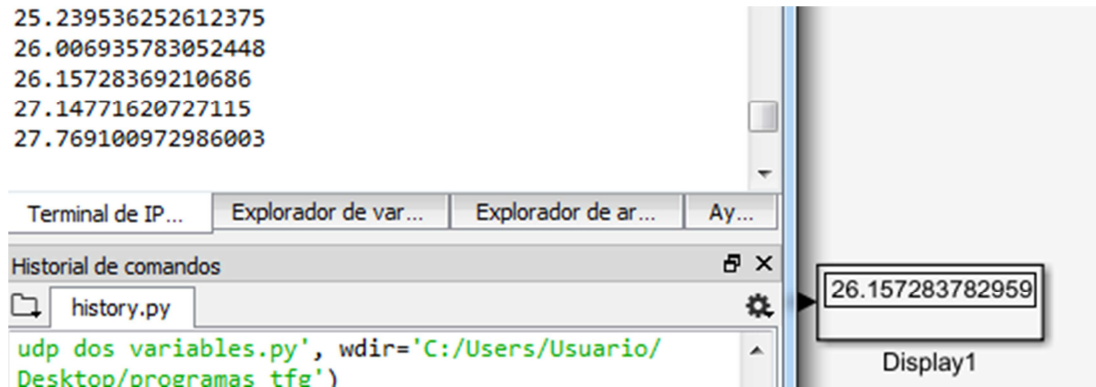


Figura 28. Primer ejemplo de precisión en la transmisión de datos.

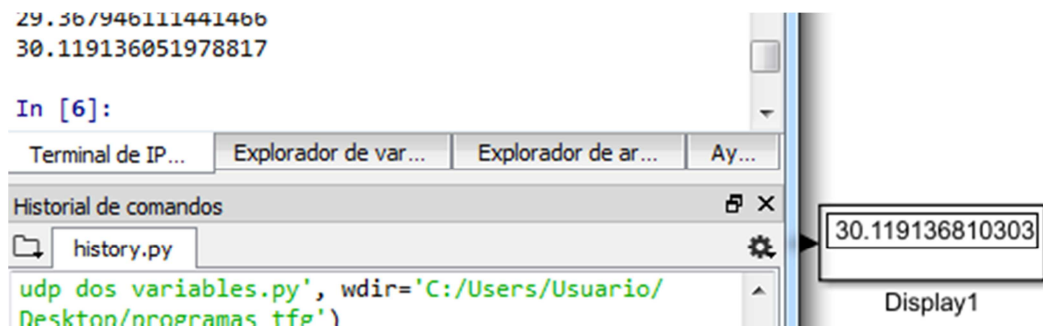


Figura 29. Segundo ejemplo de precisión en la transmisión de datos.

Se modifica el sistema de control del apartado anterior para introducir en la realimentación de la posición en los ejes X e Y las coordenadas obtenidas en el programa de visión de la marca objetivo. Esta operación implica la inutilización de los *encoder* que miden la posición en los ejes X e Y, y en su lugar se utiliza el bloque 'UDP receive' de Matlab/Simulink para conocer dichas coordenadas de la posición de la carga, tal como se observa en la figura 30.

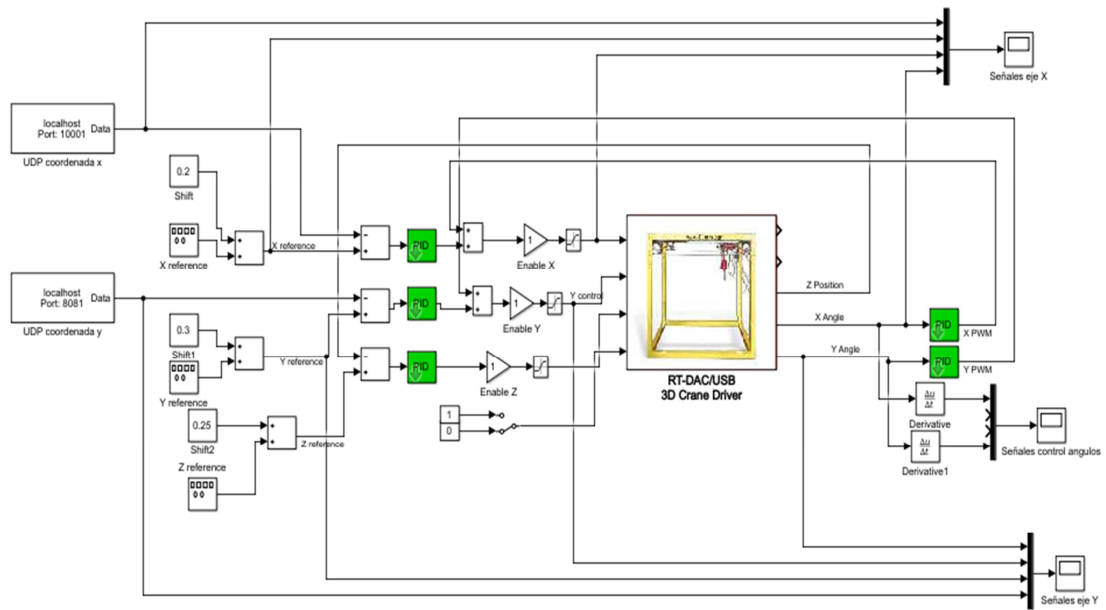


Figura 30. Sistema de control modificado.

5. Resultados

En este apartado, se explican los resultados obtenidos con nuestro sistema de realimentación por videocámara de la posición de la carga al sistema de control del puente grúa.

5.1. Resultados de rapidez de la aplicación

La captura de instantáneas mediante grabación de video supone un gran avance respecto a la captura de instantáneas directamente, ya que rebaja el tiempo necesario para esta tarea de 2.983 segundos a un tiempo casi imperceptible.

Con el fin de observar la rapidez en la transmisión de datos de Spyder a Matlab, se simulan posibles trayectorias seguidas por el puente grúa creando dos programas en Spyder que envían las coordenadas de posición de las trayectorias cada 0.1 segundos. Se observa que el tiempo desde que Spyder envía las coordenadas hasta que Matlab-Simulink las recibe es igual a 0.2 segundos. A continuación se explican los programas y los resultados obtenidos en su ejecución.

El primer programa, disponible en el anexo 5, simula la trayectoria de la carga desde el origen del puente grúa hasta las coordenadas del centro (460,460). Las coordenadas se envían a Simulink cada 0.1 segundos y se representan en tres osciloscopios como se puede ver en la figura 31. El primer osciloscopio representa la coordenada X frente al tiempo, mostrándose en la figura 32, el segundo osciloscopio representa la coordenada Y frente al tiempo y el resultado se puede ver en la figura 33, y el tercer osciloscopio, la coordenada X frente a la coordenada Y. El resultado del último osciloscopio se puede observar en la figura 34

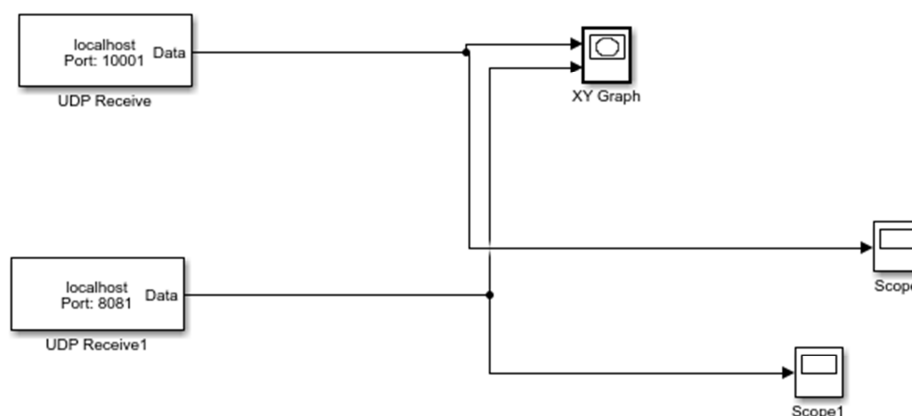


Figura 31. Esquema de los receptores UDP y los osciloscopios en los programas de prueba.

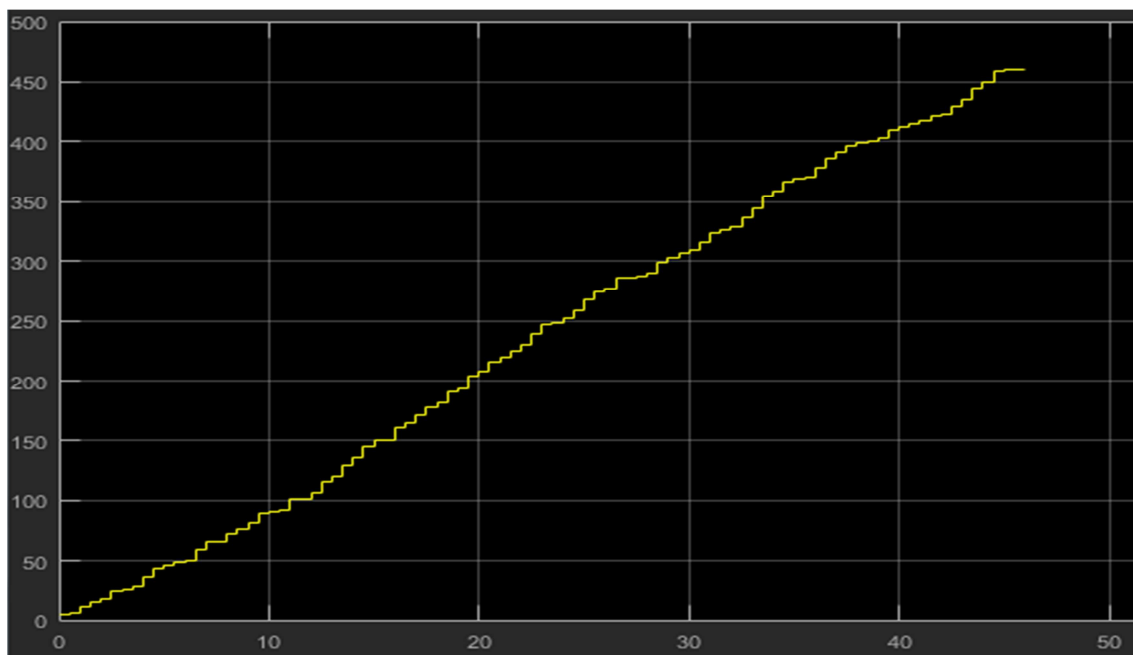


Figura 32. Representación OX-t del primer programa de simulación.

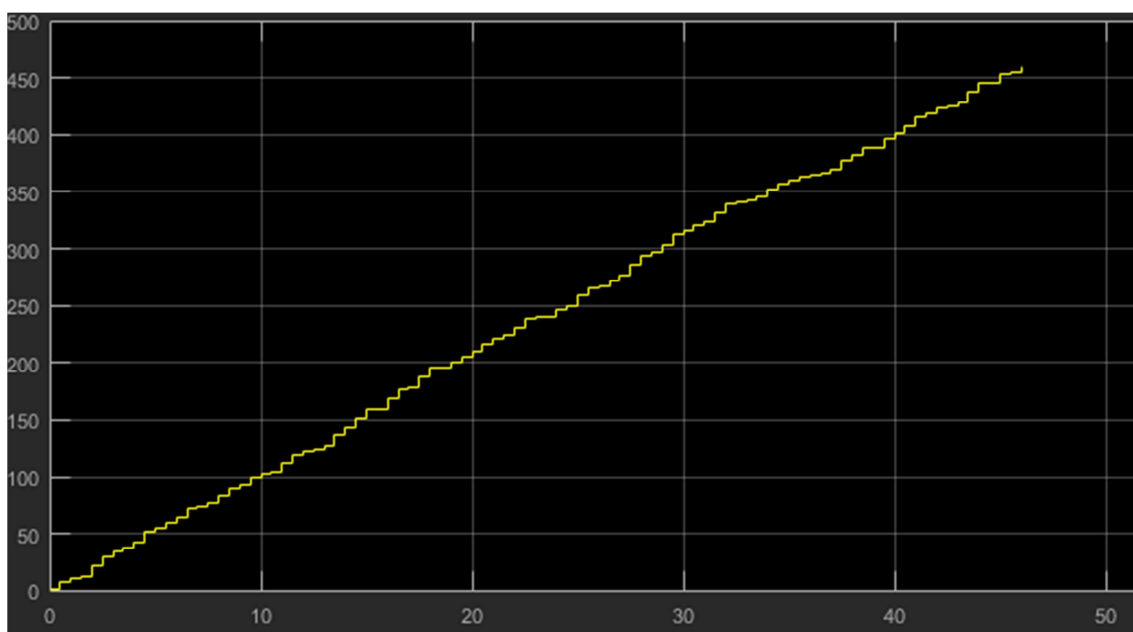


Figura 33. Representación OY-t del primer programa de simulación.

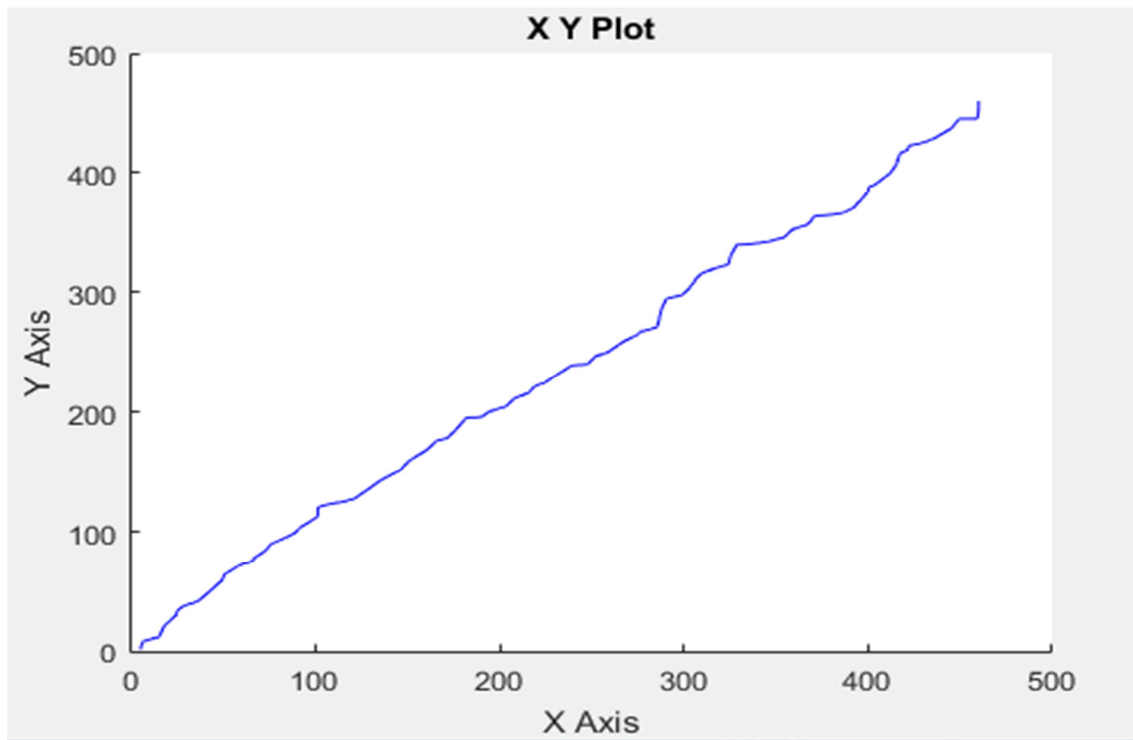


Figura 34. Representación OY-OX del primer programa de simulación.

El segundo programa, también disponible en el anexo 5, simula la trayectoria de la carga del puente grúa cuando se desplaza desde el origen, por los bordes de su espacio de trabajo, por todas las esquinas de dicho espacio de trabajo. El primer movimiento se realiza a lo largo del eje X, desde el origen (0,0) hasta llegar a la posición de la siguiente esquina (920,0). Después se desplaza a la posición de la esquina opuesta al origen (920,920). Posteriormente, se desplaza a la posición de la esquina restante (0,920), y por último realiza un desplazamiento de esta esquina al origen.

La representación del movimiento de la carga en el eje X frente al tiempo se puede ver en la figura 35 mientras que la del eje Y frente al tiempo se puede observar en la figura 36. Por último, la figura 37 representa el movimiento del eje X frente al eje Y.

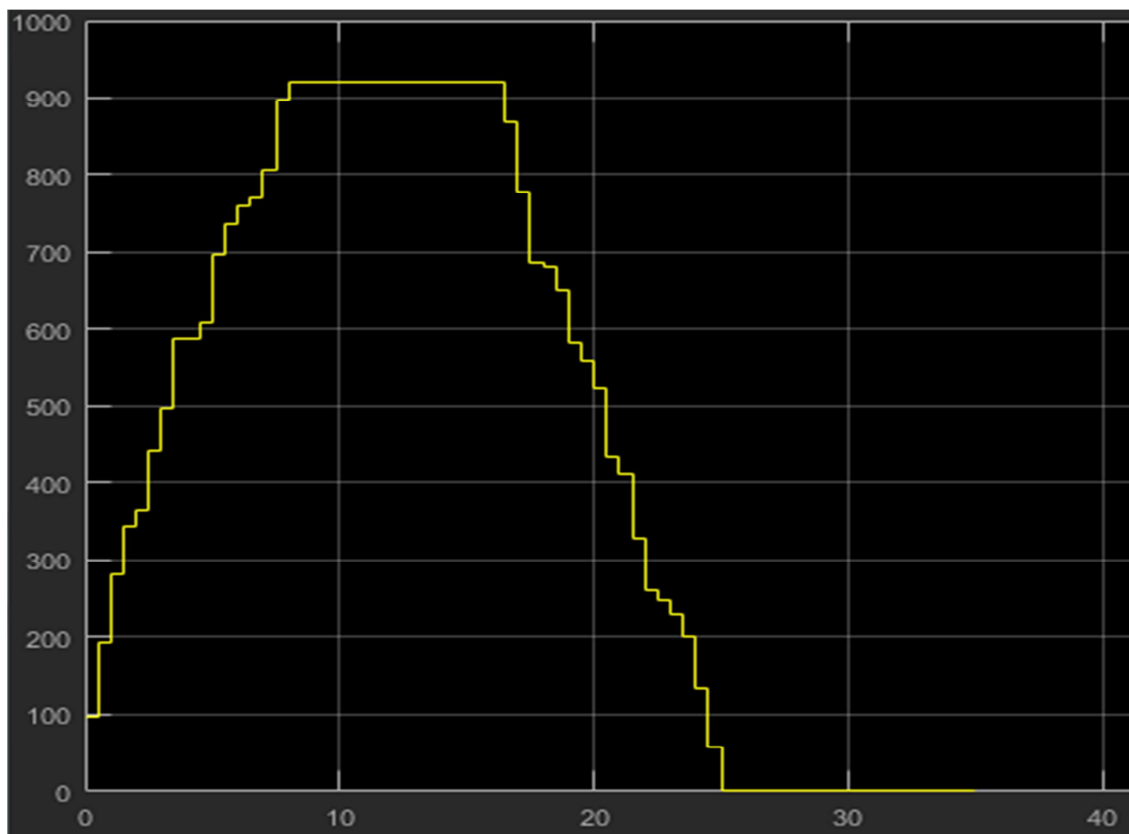


Figura 35. Representación OX-t del segundo programa de simulación.

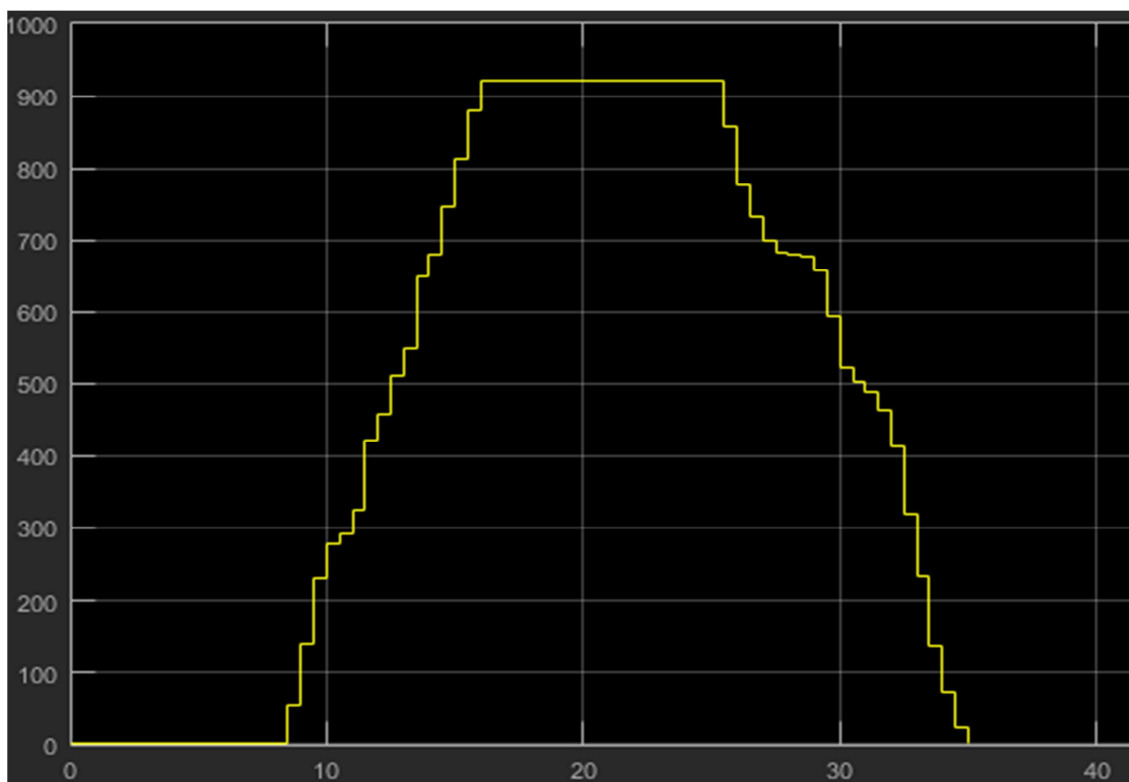


Figura 36. Representación OY-t del segundo programa de simulación.

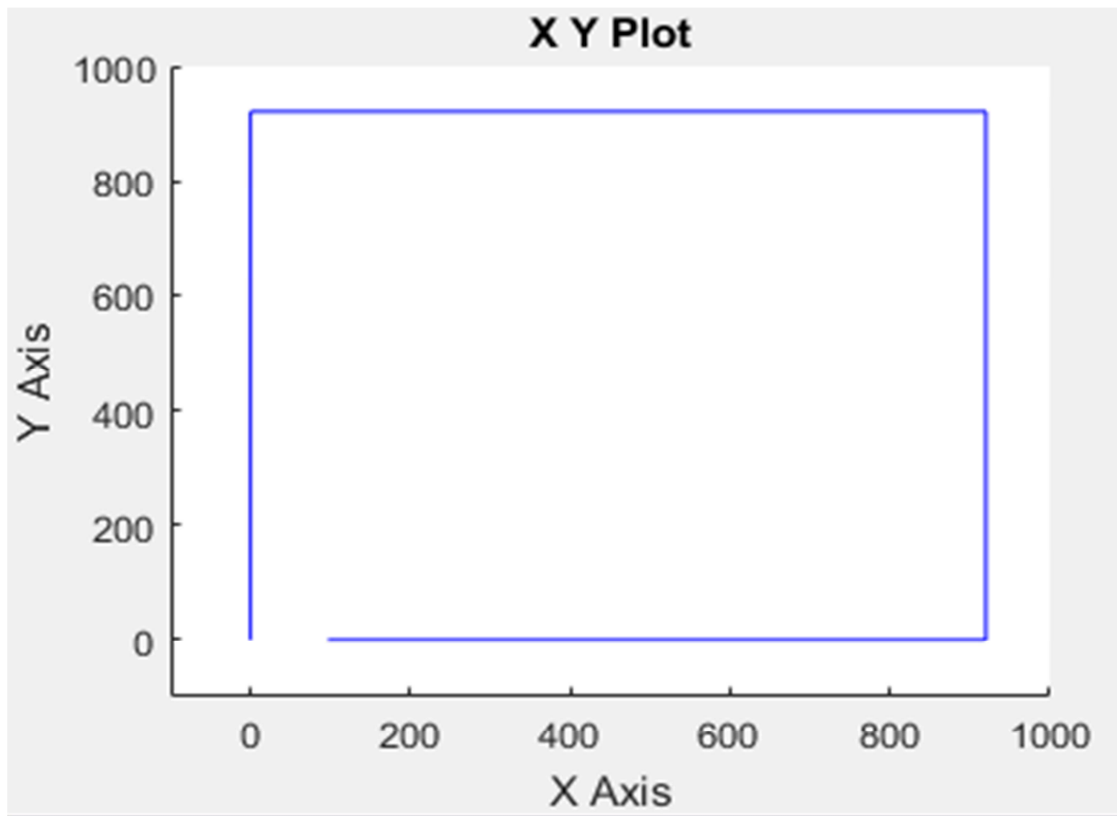


Figura 37. Representación OY-OX del segundo programa de simulación.

5.2. Resultados de precisión de la aplicación

Como se puede ver en la figura 4, la posición real de la carga, que se puede observar por la proyección de la sombra de esta sobre el suelo, difiere mucho de la posición que observa la videocámara, lo que ha obligado a utilizar infinitud de veces la seta de emergencia del puente grúa para evitar fuertes choques del carril y del carro contra los topes situados en los límites del espacio de trabajo de los mismos. Esta imprecisión se acentúa a medida que se eleva la carga, ya que a cierta elevación, la videocámara observa la marca fuera del espacio de trabajo. Se estaba estudiando este inconveniente con el fin de observar una posición más real desde la videocámara, pero debido a la pandemia actual de SARS-COV2 y a la caducidad de la tarjeta de acceso al laboratorio ha sido imposible seguir con esta investigación por falta de tiempo debido a los diversos retrasos sufridos por los motivos anteriores. Por lo tanto se concluye que este sistema de realimentación es más impreciso que los sistemas de realimentación por codificadores rotatorios y por modelado matemático de representación de espacio de estados.

6. Conclusiones y propuestas de mejora

En este apartado se describen las conclusiones del proyecto y se desarrollan propuestas que pueden ayudar a realizar un sistema de control robusto partiendo del trabajo realizado en este proyecto.

6.2. Conclusiones

La conclusión principal a la que se llega es la imprecisión de nuestro sistema de realimentación frente al sistema original de realimentación (Utilizando codificadores rotatorios) y frente al sistema de realimentación por modelado matemático de representación de espacio de estados. Esto se debe a las limitaciones que ofrece la propia videocámara y a la ubicación de esta, ya que desde la mejor posición posible de la videocámara para observar todas las marcas, se tiene una visión frontal de un plano girado cierto ángulo respecto al plano horizontal, lo que supone la obtención de las coordenadas de una posición irreal de la carga.

Una segunda conclusión a la que se ha llegado, es la rapidez de ejecución de la aplicación de Spyder y de la transmisión con gran precisión de coordenadas de Spyder a Matlab-Simulink. Este logro permite realizar en Spyder programas de generación de trayectorias en el espacio de trabajo del puente grúa y enviarlas a la entrada del sistema de control del mismo a través de la comunicación que se ha creado con Matlab-Simulink, lo que supondría la simulación de transporte de cargas siguiendo distintas trayectorias. Además, podría suponer una mejora en la aplicación de la tesis de control del puente grúa con visión artificial, ya que se podría simular el espacio de trabajo del puente grúa, sin obstáculos que impidieran colocar la cámara en la vertical del centro del cuadrado generado por las cuatro marcas de referencia, y manualmente realizar trayectorias con la marca objetivo en este espacio de trabajo simulado para que posteriormente el puente grúa realizara las mismas trayectorias sobre el espacio de trabajo real.

6.3. Propuestas de mejora

En el transcurso del proyecto se han resuelto diversos problemas, pero ha habido una serie de inconvenientes imposibles de solucionar con el equipo de trabajo y con las condiciones del laboratorio donde se sitúa el puente grúa, debido a la falta de tiempo que ha ocasionado la pandemia de SARS-COV2. Por estos motivos, se desarrollan en este apartado una serie de mejoras que se ha pensado que pueden sumar funcionalidad y precisión a la aplicación creada.

El primer inconveniente al que se le propone mejora que es proporcionar una iluminación dentro de un intervalo constante en la sala donde se instala el puente grúa. Esto evita una detección ineficaz de las marcas en determinadas situaciones, y por lo tanto ahorra tiempo en el ajuste de parámetros para su detección.

El segundo y principal inconveniente es el tema de la posición de la videocámara, para el cual se pensaron tres alternativas:

La primera alternativa fue colocarla por encima del puente grúa, pero rápidamente se observó que el propio carro del puente grúa obstaculizaba la visión de la carga en demasiadas situaciones como para elegir esta alternativa para nuestra aplicación.

La segunda alternativa fue colocarla debajo del carro, pero se comprobó posteriormente que frente a movimientos bruscos de la videocámara, la aplicación no responde y es necesario un reinicio de Spyder. Este hecho obligó a descartar esta alternativa.

La tercera alternativa fue la que se escogió para el desarrollo del presente proyecto y se ha descrito en apartados anteriores, pero la puesta en marcha del sistema ha supuesto un esfuerzo por posicionar la videocámara de tal forma que se reconozcan los centros de las cinco marcas.

Esta situación se podría mejorar con una cámara que tenga un sistema de visión más robusto frente a movimientos. Este tipo de cámara pudiera ser cualquier cámara que de buenos resultados en sistemas dinámicos, por ejemplo cámaras insertadas en objetos móviles o videocámaras deportivas. Y si una cámara de este tipo pudiera colocarse debajo del carro del puente grúa, sería necesario modificar el parámetro de la masa del carro, pero se confía en que sería la solución que mejores resultados proporcionaría.

Para terminar, el último obstáculo que se ha tenido que superar ha sido la orientación del puente grúa. Se podría solucionar con una rotación de 180° del puente grúa, pero actualmente es muy difícil debido a la ubicación de elementos necesarios para el funcionamiento del mismo, así como por el espacio disponible en el laboratorio para la instalación del puente grúa.

Referencias

- [1]. Definición de grúa, Wikipedia. [Online]. Disponible: [https://es.wikipedia.org/wiki/Gr%C3%BAA_\(m%C3%A1quina\)](https://es.wikipedia.org/wiki/Gr%C3%BAA_(m%C3%A1quina))
- [2]. Definición de puente grúa, Wikipedia. [Online]. Disponible: <https://es.wikipedia.org/wiki/Puente-gr%C3%BAA>
- [3]. Definición de sistema de control, Wikipedia. [Online]. Disponible: https://es.wikipedia.org/wiki/Sistema_de_control
- [4]. Puente grúa de INTECO. [Online]. Disponible: <http://www.inteco.com.pl/products/3d-crane>
- [5]. RT-DAC/USB2 I/O Board (Board Version 1.03) , User's Manual, INTECO, Kraków, Polonia, 2010
- [6]. INTECO 3D Crane - Installation Manual, INTECO, Kraków, Polonia, 2010
- [7]. INTECO 3D Crane – User's Manual, INTECO. [Online]. Disponible: <http://a-lab.ee/man/3DCrane-user-manual.pdf>
- [8]. R. Louk, “Computer Vision based Control System for 3DCrane”, Bachelor's thesis, Department of Computer Control, Faculty of Information Technology, Tallinn University of Technology, Tallinn, Harju, Estonia, 2016
- [9]. A. Aksjonov, “3D KRAANA JUHTIMISSÜSTEEM”, Magistritöö, Institute of Electrical Engineering, Tallinn University of Technology, Tallinn, Harju, Estonia, 2015
- [10]. Definición de Matlab, Wikipedia. [Online]. Disponible: <https://es.wikipedia.org/wiki/MATLAB>
- [11]. Definición de Simulink, Wikipedia. [Online]. Disponible: http://isa.uniovi.es/~arobles/AyC/S1_simulink.pdf
- [12]. Real-Time Windows Target, User's Guide Version 2, The Mathworks. [Online]. Disponible: http://users.isr.ist.utl.pt/~alex/micd0405/RTWT_user_guide.pdf
- [13] Definición de Python, Diccionario de informática y tecnología, ALEGSA, Santa Fe, Argentina. [Online]. Disponible: <https://www.alegsa.com.ar/Dic/python.php>
- [14] D. Naranjo. OpenCV una biblioteca para el reconocimiento de objetos en imágenes y cámaras, DesdeLinux. [Online]. Disponible: <https://blog.desdelinux.net/opencv-una-biblioteca-para-el-reconocimiento-de-objetos-en-imagenes-y-camaras/>
- [15]. Definición de Anaconda, Wikipedia. [Online]. Disponible: [https://es.wikipedia.org/wiki/Anaconda_\(distribuci%C3%B3n_de_Python\)](https://es.wikipedia.org/wiki/Anaconda_(distribuci%C3%B3n_de_Python))

- [16]. Damián A., Spyder, un potente entorno de desarrollo interactivo para Python, Ubunlog. [Online]. Disponible: <https://ubunlog.com/spyder-entorno-desarrollo-python/>
- [17]. Visión artificial con Open CV, Unipython. [Online]. Disponible: <https://unipython.com/>
- [18]. Gl4r3, Detección de colores con OpenCV y Python, Robologs. [Online]. Disponible: <https://robologs.net/2016/05/18/detectar-multiples-colores-con-opencv-y-python/>
- [19]. Emulador de Puerto Serie, Eltima Software. [Online]. Disponible: <https://www.virtual-serial-port.org/es/products/virtual-com-port-emulator/>
- [20]. UDP: ¿qué es el protocolo UDP?, Ionos Digital Guide, Ionos. [Online]. Disponible: <https://www.ionos.es/digitalguide/servidores/know-how/udp-user-datagram-protocol/>
- [21]. UDP Receive, The MathWorks. [Online]. Disponible: <https://es.mathworks.com/help/instrument/udpreceive.html>
- [22]. Instalación de un puente grúa, Logsteel. [Online]. <https://www.logsteel.es/instalacion-puente-grua-demag-20-tns/>