INSTITUTO POLITECNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA DE INGENIERÍA Y TECNOLOGIAS AVANZADAS

Trabajo Terminal II

"Prototipo de sistema para seguimiento de objetos con visión artificial"

Qué para obtener el título de

"Ingeniero en Mecatrónica"

Presenta el alumno

Luis Alberto García Barajas

Asesores

Dr. Juan Luis Mata Machuca

Dr. Alberto Luviano Juárez

Dr. Leonel Germán Corona Ramírez

Marzo 2015

upiita·ipn

INSTITUTO POLITECNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA DE INGENIERÍA Y TECNOLOGIAS AVANZADAS

Trabajo Terminal II

"Prototipo de sistema para seguimiento de objetos con visión artificial"

Qué para obtener el título de

"Ingeniero en Mecatrónica"

Presenta el alumno

Luis Alberto García Barajas

Asesores

Dr. Juan Luis Mata Machuca — **Dr. Alberto Luviano Juárez**

Dr. Leonel Germán Corona Ramírez

Presidente del Jurado — *Profesor Titular*

M en C. Griselda Stephany Abarca Jiménez — **Ing. Omar Heredia Vargas**

upiita·ipn

Índice

Índice de Figuras

Índice de tablas

Resumen

Este trabajo se enfoca en el desarrollo e integración de un prototipo robótico de tres grados de libertad que implemente visión artificial, que sea capaz de realizar el seguimiento de la trayectoria de un objeto determinado. El sistema cuenta con los subsistemas de visión artificial, subsistema robótico y subsistema de control, los cuales implementados dentro del sistema completo realizan el seguimiento de la posición del objeto a detectar. El sistema robótico contara con un mecanismo que brinde desplazamiento lineal al robot y un mecanismo de dos grados de libertad para el giro de la cámara en dos ejes, logrando así un sistema de tres grados de libertad.

Abstract

This work focuses on the development and integration of a three degrees of freedom robot, to implement artificial vision, able to track the trajectory of an object. The system has the following subsystems, artificial vision subsystem, robotic subsystem and control subsystem, the complete system gets the position of the object to be detected. The robotic system has a linear mechanism displacement, and a mechanism of two grades of liberty for turning the camera in two axes, thus achieving a system of three degrees of freedom.

Palabras clave

Visión Artificial, Robot, Tracking, Objeto, Control, mecanismo, servomotor, sensor de visión.

Capítulo 1
Introducción

1. Introducción

Hoy en día la tecnología avanza a pasos agigantados siendo un soporte confiable en distintas áreas del desarrollo humano, tales como la industria, la medicina y la rama energética, pero también en otras áreas de menor responsabilidad social como son la industria del entretenimiento, los deportes y las redes sociales, es por ello que en este trabajo se espera brindar una muestra de este avance de la tecnología que tendrá aplicación en cualquiera de las mencionadas áreas.

La cinematografía no solo puede representar un objeto en movimiento sobre un escenario, si no que puede usar una cámara que representa el punto de vista o perspectiva de la audiencia, que se mueve durante la filmación. El movimiento juega un rol considerable en el lenguaje emocional de la filmación de imágenes y la reacción emocional de la audiencia a la acción. Las técnicas se extienden desde los movimientos más básicos del "panning" (movimiento horizontal de un punto de vista desde una posición fija; como si giraras tu cabeza de un lado a otro) y "tilting" (movimiento vertical de un punto de vista desde una posición fija; como inclinar la cabeza hacia atrás para mirar hacia el cielo y luego hacia el suelo) hasta el"dollying" (posicionando la cámara en una plataforma en movimiento para acercarse o alejarse del objeto). (Wheeler, 2006)

Para ello se plantea el desarrollo e integración de un sistema de visión artificial con un sistema robótico para el seguimiento de la trayectoria de un objeto que cubra las necesidades que lleguen a surgir en cualquier área.

1.1. Planteamiento del problema

La industria cinematográfica está constantemente evolucionando en la forma de realizar películas, cada vez se implementa el uso de tecnologías avanzadas en aspectos de efectos especiales, tecnología digital, tecnologías de grabación, etc. Las técnicas de grabación y movimientos de la cámara no se quedan atrás,

Las cámaras se han llegado a montar en casi cualquier medio de transporte imaginable; la mayoría de las cámaras son de mano, eso quiere decir que están en manos del camarógrafo, que la mueve de una posición a otra mientras filman la acción.

El problema se ha tomado de las técnicas de grabación existentes, ya que en los estudios de grabación es un trabajo complicado mover todo un sistema de cámaras para darle seguimiento a la trayectoria de un objeto, en este caso el objeto puede ser un actor o un móvil, en ocasiones el espacio donde deben realizarse las tomas es limitado y el camarógrafo no puede llegar a realizar la toma correctamente, además del elevado costo que este equipo representa y la inexactitud de los operarios.

Figura 1 Sistema de grabación dollying

Es por ello que se propone el diseño, construcción y control de un prototipo de sistema de visión artificial que realice el seguimiento de un objeto a través de un sistema robótico de tres grados de libertad.

1.2. Objetivo General

Diseño, construcción y control de un prototipo de robot de tres grados de libertad con un sistema de visión artificial que detecte el movimiento de un objeto determinado y realice el seguimiento del mismo.

1.3. Objetivos Particulares

- Implementación de un sistema de reconocimiento para un objeto determinado en un ambiente controlado a través de algoritmos de visión artificial, utilizando herramientas computacionales.
- Diseño y construcción de un mecanismo de riel que brinde desplazamiento lineal al robot (un grado de libertad).
- Diseño y construcción de un mecanismo de dos grados de libertad para el movimiento de una cámara.
- Implementación de un sistema de control para lograr la conjunción del sistema de visión artificial y el sistema robótico de tres grados de libertad.

1.4. Justificación

Uno de los propósitos fundamentales en el campo de la robótica es crear robots autónomos, tales robots aceptarán tareas descritas, que deberán ejecutarlas sin intervención humana posterior. Los sistemas de percepción han sido utilizados en una gran variedad de problemas relacionados con la robótica. Debido a la poca competencia en el mercado y altos costos que manejan empresas dedicadas a la grabación cinematográfica y eventos deportivos, existe la necesidad de desarrollar un sistema de seguimiento de objetos de 3 grados de libertad mediante visión artificial.

El dispositivo que se realizará contará con sistemas de visión, esta característica no está implementada en sistemas comerciales en la actualidad, por lo que es una innovación a los sistemas que actualmente se usan. Esta implementación tecnológica ayudará a obtener capturas difíciles de realizar y simplificará el tiempo de grabación considerablemente. Este prototipo sirve como plataforma para que en un futuro sea aplicado de acuerdo a las necesidades del sector donde lo requieran, así como para realizar futuras innovaciones, además de realizar un prototipo interdisciplinario que cumpla con diferentes áreas de conocimiento integradas de manera sinérgica, cumpliendo con el concepto de Ingeniería Mecatrónica.

Capítulo 2
Marco Teórico

2. Marco teórico

2.1. Visión Artificial

2.1.1. Introducción

Podríamos decir que la Visión Artificial describe la deducción automática de la estructura y propiedades de un mundo tridimensional posiblemente dinámico, bien a partir de una o varias imágenes bidimensionales del mundo. Las imágenes pueden ser monocromáticas (de niveles de gris) o colores, pueden provenir de una o varias cámaras e incluso cada cámara puede estar estacionaria o móvil.

Las estructuras y propiedades del mundo tridimensional que queremos deducir en visión artificial incluyen no sólo sus propiedades geométricas, sino también sus propiedades materiales. Ejemplos de propiedades geométricas son la forma, tamaño y localización de los objetos. Ejemplos de propiedades de los materiales son su color, iluminación, textura y composición. Si el mundo se modifica en el proceso de formación de la imagen, necesitaremos inferir también la naturaleza del cambio, e incluso predecir el futuro.

La entrada a un sistema de Visión Artificial es una imagen obtenida por un elemento de adquisición, mientras que su salida es una descripción de la escena, la cual ha sido obtenida a partir de la imagen. Por un lado, esta descripción debe estar relacionada de algún modo con aquella realidad que produce la imagen y, por el otro, debe contener toda la información requerida para la tarea de interacción con el medio ambiente que se desea llevar a cabo, por ejemplo mediante un robot. Esto es, la descripción depende en alguna forma de la entrada visual y debe proporcionar información relevante y utilizable por el robot. (Azuela, 2013)

La visión, tanto para un hombre como para un ordenador, consta principalmente de dos fases: captar una imagen e interpretarla. A pesar de la complejidad que presenta el ojo humano, la fase de captación de imágenes hace mucho tiempo que está resuelta. El ojo del ordenador es la cámara de video, y su retina un sensor que es sensible a la intensidad luminosa. Así que en la visión artificial lo que resta es interpretar las imágenes, distinguir los objetos de la escena, extraer información de ellos y resolver aspectos más particulares según las necesidades que se deseen satisfacer.

Las tareas de pre procesamiento consisten en algoritmos matemáticos que calculan nuevas intensidades luminosas para los pixeles y la mayoría, aunque sencillos, consumen gran cantidad de tiempo de cálculo, tanto mayor cuanta más resolución tenga la imagen. Esto representa un inconveniente cuando se pretende que el ordenador sea capaz de ver en tiempo real, es decir, que responda de forma casi instantánea a como se producen las imágenes frente a sus ojos. Para que un sistema funcione en tiempo real es necesario estudiar todos los métodos posibles y la forma de realizarlos con la menos carga computacional posible. (Basu, 1993)

2.1.2. Aplicaciones

El amplio espectro de aplicaciones cubierto por la Visión Artificial, se debe a que permite extraer y analizar información espectral, espacial y temporal de los distintos objetos.

La información espectral incluye frecuencia (color) e intensidad (tonos de gris). La información espacial se refiere a aspectos como forma y posición (una, dos y tres dimensiones). La información temporal comprende aspectos estacionarios (presencia y/o ausencia) y dependientes del tiempo (eventos, movimientos, procesos). La mayoría de las aplicaciones de la visión artificial podemos clasificarlas por el tipo de tarea, entre las que mencionaremos a continuación. (Marcos, 2006)

- **La medición o calibración:** se refiere a la correlación cuantitativa con los datos del diseño, asegurando que las mediciones cumplan con las especificaciones del diseño. Por ejemplo, el comprobar que un cable tenga el espesor recomendado.

- **La detección de fallas:** es un análisis cualitativo que involucra la detección de defectos o artefactos no deseados, con forma desconocida en una posición desconocida. Por ejemplo, encontrar defectos en la pintura de un auto nuevo, o agujeros en hojas de papel.

- **La verificación:** es el chequeo cualitativo de que una operación de ensamblaje ha sido llevada a cabo correctamente. Por ejemplo, que no falte ninguna tecla en un teclado, o que no falten componentes en un circuito impreso.

- **El reconocimiento:** involucra la identificación de un objeto con base en descriptores asociados con el objeto. Por ejemplo, la clasificación de cítricos (limones, naranjas, mandarinas, etc.) por color y tamaño.

- **La identificación:** es el proceso de identificar un objeto por el uso de símbolos en el mismo. Por ejemplo, el código de barras, o códigos de perforaciones empleados para distinguir hule de espuma de asientos automotrices.

- **El análisis de localización:** es la evaluación de la posición de un objeto. Por ejemplo, determinar la posición donde debe insertarse un circuito integrado.

- **Guía:** significa proporcionar adaptativamente información posicional de retroalimentación para dirigir una actividad. El ejemplo típico es el uso de un Sistema de Visión para guiar un brazo Robótico mientras suelda o manipula partes. Otro ejemplo sería la navegación en vehículos autónomos.

2.1.3. Seguimiento de objetos

El seguimiento de objetos es el proceso de estimar en el tiempo la ubicación de uno o más objetos móviles mediante el uso de una cámara. La rápida mejora en cuanto a calidad y resolución de los sensores de imagen, juntamente con el dramático incremento en cuanto a la potencia de cálculo en la última década, ha favorecido la creación de nuevos algoritmos y aplicaciones mediante el seguimiento de objetos.

El seguimiento de objetos puede ser un proceso lento debido a la gran cantidad de datos que contiene un video. Además, la posible necesidad de utilizar técnicas de reconocimiento de objetos para realizar el seguimiento incrementa su complejidad. (Lu, 2001)

2.1.4. Dificultades del seguimiento

Los principales retos que hay que tener en cuenta en el diseño de un seguidor de objetos están relacionados con la similitud de aspecto entre el objeto de interés y el resto de objetos en la escena, así como la variación de aspecto del propio objeto. Dado que el aspecto tanto del resto de objetos como el fondo puede ser similar al del objeto de interés, esto puede interferir en su observación. En ese caso, las características extraídas de esas áreas no deseadas puede ser difícil de diferenciar de las que se espera que el objeto de interés genere. Este fenómeno se conoce con el nombre de clutter. (A. Yilmaz, 2006)

Además del reto de seguimiento que causa el clutter, los cambios de aspecto del objeto en el plano de la imagen dificultan el seguimiento causado por uno o más de los siguientes factores:

- **Cambios de posición:** El objeto móvil de interés varía su aspecto cuando se proyecta sobre el plano de la imagen, por ejemplo, al girar.

- **Iluminación ambiente:** La dirección, la intensidad y el color de la luz de ambiente influyen en el aspecto del objeto de interés. Asimismo, los cambios en la iluminación global son con frecuencia un reto en las escenas al aire libre.

- **Ruido:** El proceso de adquisición de imágenes introduce en la señal de la imagen un cierto grado de ruido que depende de la calidad del sensor. Las observaciones del objeto de interés pueden dañarse y por tanto afectar al rendimiento del seguidor.

- **Oclusiones:** Puede ser que un objeto de interés no se observe bien cuando sea parcial o totalmente tapado por otros objetos en la escena.

2.1.5. Representación del objeto

En un escenario de seguimiento, un objeto se puede definir como cualquier cosa que sea de interés para su posterior análisis. Los objetos se pueden representar mediante sus formas y apariencias. A continuación se describen las representaciones de forma del objeto utilizadas generalmente: (A. Yilmaz, 2006)

- **Puntos:** El objeto está representado por un punto, es decir, por un centroide o por un conjunto de puntos. En general, la representación de puntos es adecuada para el seguimiento de objetos que ocupan pequeñas regiones en una imagen.

- **Formas geométricas primitivas:** La forma del objeto se representa con un rectángulo, una elipse, etc. El movimiento de estas representaciones es generalmente modelada por la translación, afinidad o la transformación proyectiva (homografía). Aunque las formas geométricas primitivas son más adecuadas para la representación de objetos rígidos simples, también se utiliza para el seguimiento de objetos no rígidos.

- **Silueta del objeto y contorno:** La representación del contorno define el límite de un objeto. La región dentro del contorno se conoce como la silueta del objeto. Estas representaciones son adecuadas para el seguimiento de formas complejas no rígidas.

- **Modelos articulados de forma:** Los objetos articulados están formados por partes del cuerpo que están unidas por articulaciones. Por ejemplo, el cuerpo humano es un objeto articulado por el torso, piernas, manos, cabeza y pies unidos por articulaciones. La relación entre estas partes se rige por los modelos del movimiento cinemático. Para representar un objeto articulado, se puede modelar los componentes utilizando cilindros o elipses.

- **Modelos esqueléticos:** El esqueleto del objeto se puede extraer mediante la transformación del eje medio de la silueta del objeto. La representación del esqueleto se puede utilizar para modelar objetos articulados y rígidos.

2.1.6. Selección de características

Seleccionar las características adecuadas tiene un papel fundamental en el seguimiento. En general, la característica visual más deseada es la singularidad porque los objetos se pueden distinguir fácilmente en el espacio de características. Los detalles de las características más comunes son los siguientes: (A. Yilmaz, 2006)

Color: En el procesamiento de imágenes se utiliza normalmente el espacio de color RGB (rojo, verde y azul) para representar esta característica. A pesar de ello, el espacio RGB no es un espacio de color percentualmente uniforme, y por tanto se han utilizado una gran variedad de espacios de color en el seguimiento. El color aparente de un objeto se ve influenciado principalmente por dos factores físicos:

- La distribución de energía espectral de la fuente.
- Las propiedades de reflectancia de la superficie del objeto.

Márgenes: Los límites de los objetos suelen generar fuertes cambios en la intensidad de la imagen. La detección de márgenes se utiliza para identificar dichos cambios. Una propiedad importante de los márgenes es que son menos sensibles a los cambios de iluminación en comparación con las características de color. Los algoritmos que hacen un seguimiento de los límites de los objetos suelen utilizar los márgenes como características representativas.

Flujo óptico: EL flujo óptico es un campo denso de desplazamiento de vectores que define la translación de cada píxel en una región. Se calcula mediante la restricción de brillantez constante y se utiliza generalmente como característica de la segmentación basada en movimiento, así como en aplicaciones de seguimiento.

Textura: La textura es una medida de la variación de intensidad de una superficie que cuantifica las propiedades como por ejemplo la suavidad y la regularidad. En comparación con el color, la textura requiere una etapa de procesamiento para generar los descriptores. Similares a las características de límites, las características de textura son menos sensibles a los cambios de iluminación en comparación con el color.

2.2. OpenCV

OpenCV (Open Source Computer Vision Library) es una fuente abierta de visión por computador y biblioteca de software de aprendizaje automático. OpenCV fue construido para proporcionar una infraestructura común para aplicaciones de visión por computador y para acelerar el uso de la percepción de la máquina en los productos comerciales. Al ser un producto con licencia BSD, OpenCV hace que sea fácil para las empresas a utilizar y modificar el código.

La biblioteca cuenta con más de 2500 algoritmos optimizados, lo que incluye un amplio conjunto de tanto clásico y algoritmos de visión por computador y aprendizaje automático con tecnología de última generación. Estos algoritmos se pueden utilizar para detectar y reconocer rostros, identificar objetos, clasificar las acciones humanas en videos, movimientos de cámara, objetos de pista en movimiento, extraer modelos 3D de objetos, producir nubes de puntos 3D a partir de cámaras estéreo, unir imágenes para producir una alta resolución imagen de una escena completa, encontrar imágenes similares de una base de datos de imágenes, eliminar los ojos rojos de las imágenes tomadas con flash, seguir los movimientos de los ojos, reconocer paisajes y establecer marcadores para superponer con la realidad aumentada, etc. OpenCV tiene más de 47 mil usuarios y el número estimado de descargas son superiores a 7 millones. La biblioteca se utiliza ampliamente en las empresas, grupos de investigación y por los organismos gubernamentales.

Junto con las empresas bien establecidas como Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota, que emplean la biblioteca, hay muchas nuevas empresas como Applied Minds, VideoSurf, y Zeitera, que hacen un amplio uso de OpenCV. Usos desplegados de OpenCV abarcan todo el mundo, la detección de intrusiones en video de vigilancia en Israel, el seguimiento equipamiento minero en China, ayudando a los robots navegar y recoger objetos en Willow Garage, detección de accidentes de piscina por ahogamiento en Europa, corriendo el arte interactivo en España y Nueva York, comprobando pistas de escombros en Turquía, la inspección de las etiquetas de los productos en las fábricas de todo el mundo a la detección de rostros rápida en Japón.

Cuenta con interfaces de C ++, C, Python, Java y Matlab y es compatible con Windows, Linux, Android y Mac OS. OpenCV se inclina principalmente hacia las aplicaciones de visión en tiempo real y se aprovecha de instrucciones MMX y SSE cuando esté disponible. OpenCV está escrito de forma nativa en C ++ y tiene una interfaz de plantilla que funciona a la perfección con los contenedores STL. (OpenCV, http://opencv.org, 2014)

2.3. Etapas en un proceso de visión artificial

La visión artificial lleva asociada una enorme cantidad de conceptos relacionados con hardware, software y también con desarrollos teóricos.

El primer paso en el proceso es adquirir la imagen digital. Para ello se necesitan sensores y la capacidad para digitalizar la señal producida por el sensor. Una vez que la imagen digitalizada ha sido obtenida, el siguiente paso consiste en el pre procesamiento de dicha imagen. El objetivo del pre procesamiento es mejorar la imagen de forma que el objetivo final tenga mayores posibilidades de éxito.

El paso siguiente es la segmentación, su objetivo es dividir la imagen en las partes que la construyen o los objetos que la forman. En general la segmentación autónoma es uno de los problemas más difíciles en el procesamiento de la imagen. Por una parte, una buena segmentación facilitara mucho la solución del problema; por otra la segmentación errónea conducirá al fallo.

La salida del proceso de segmentación es una imagen de datos que, o bien contiene la frontera de la región o los puntos de ella misma. Es necesario convertir estos datos a una forma que sea apropiada para el ordenador. La primera decisión es saber si se va a usar la representación por frontera o región completa. La representación por la frontera es apropiada cuando el objetivo se centra en las características de la forma externa como esquinas o concavidades y convexidades. La representación por regiones es apropiada cuando la atención se centra en propiedades internas como la textura o el esqueleto.

Es necesario especificar un método que extraiga los datos de interés. La parametrización, que recibe también el nombre de selección de rasgos se dedica a extraer rasgos que producen alguna información cuantitativa de interés o rasgos que son básicos para diferenciar una clase de objetos de otra.

En último lugar se encuentra el reconocimiento y la interpretación. El reconocimiento es el proceso que asigna una etiqueta a un objeto basada en la información que proporcionan los descriptores (clasificación). La interpretación lleva a asignar significado al conjunto de objetos reconocidos.

2.3.1. Adquisición de la imagen

La Webcam es el dispositivo encargado de transformar las señales luminosas que aparecen en la escena, en señales analógicas capaces de ser transmitidas a la computadora. Se divide en dos partes, el sensor, que captura las propiedades del objeto en forma de señales luminosas y lo transforma en señales analógicas, y la óptica que se encarga de proyectar los elementos adecuados de la escena ajustando una distancia focal adecuada.

Los sensores de visión usados mas recientemente son los basados en matrices de dispositivos acoplados por carga CCD o CMOS; estos transductores proporcionan una señal con amplitud proporcional a la luminosidad de la escena y realizan una digitalización espacial completa en dos dimensiones (líneas y columnas), pues descomponen la imagen en una matriz de puntos. La codificación de la brillantez de cada elemento de imagen o pixel, obtenido de la digitalización espacial, se hace generalmente en 8 bits.

2.3.2. Métodos de Almacenamiento

Esto es acerca de cómo se almacenan los valores de los píxeles. Puede seleccionar el espacio de color y el tipo de datos utilizado. El espacio de color se refiere a cómo combinamos componentes de color con el fin de codificar un color determinado. La más simple es la escala de grises, donde los colores de los que disponemos son de color blanco y negro. La combinación de estos nos permite crear muchos tonos de gris.

Para formas coloridas tenemos muchos más métodos para elegir, cada uno de ellos se descompone a tres o cuatro componentes básicos y podemos utilizar la combinación de estos para crear los otros.

La más popular es RGB, sobre todo porque se trata también de cómo nuestro ojo acumula colores. Sus colores básicos son el rojo, verde y azul. Para codificar la transparencia de un color a veces se añade un cuarto elemento: alfa (A).

Hay, sin embargo, muchos otros sistemas de color, cada uno con sus propias ventajas:

RGB es el más común ya que nuestros ojos utilizan algo similar, nuestros sistemas de visualización también componen colores utilizando estos.

El HSV y HLS descomponen los colores en su tono, la saturación y los componentes de valor / luminancia, que es una forma más natural para nosotros de describir colores. Es posible quitar el último componente, por lo que su algoritmo será menos sensible a las condiciones de luz de la imagen de entrada. (Marcos, 2006)

2.3.3. Convertir imagen del espacio RGB a HSV

RGB es un modelo de color basado en la síntesis aditiva, con el que es posible representar un color mediante la mezcla por adición de los tres colores de luz primarios, mientras que el modelo de color HSV es la representación de coordenadas cilíndricas más común de los puntos en un modelo de color RGB, fue desarrollado en la década de 1970 para aplicaciones gráficas por computadora. (Marcos, 2006)

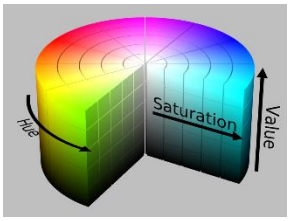


Figura 2 modelo de color HSV

La transformación de los modelos RGB al modelo HSV se realiza de la siguiente manera.

$$V \leftarrow \max(R, G, B)$$

$$S \leftarrow \begin{cases} \frac{V - \min(R,G,B)}{V} & \text{if } V \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$H \leftarrow \begin{cases} 60(G - B)/(V - \min(R, G, B)) & \text{if } V = R \\ 120 + 60(B - R)/(V - \min(R, G, B)) & \text{if } V = G \\ 240 + 60(R - G)/(V - \min(R, G, B)) & \text{if } V = B \end{cases}$$

Si $H < 0$ a continuación $H \leftarrow H + 360$. En la salida $0 \leq V \leq 1, 0 \leq S \leq 1, 0 \leq H \leq 360$.

2.3.4. Etapa de segmentación - Operaciones morfológicas

Las operaciones morfológicas son métodos para procesar imágenes binarias basadas sobre formas. Estas operaciones toman una imagen binaria como entrada y dan como resultado una imagen binaria como salida. El valor de cada pixel en la imagen de salida está basado sobre el correspondiente pixel de entrada y sus vecinos. Dentro de las operaciones morfológicas tenemos la dilatación y erosión, las cuales son implementadas dentro de nuestro algoritmo. (Marcos, 2006)

La dilatación adiciona pixeles a los límites del objeto (es decir los cambia de off a on), y la erosión remueve pixeles sobre los límites del objeto (los cambia de on a off).

Erosión binaria

Dada una imagen A se erosiona por B cuando para todos los puntos x tales que B, trasladando por x, está contenido en A, es decir, la erosión pone a cero todos los pixeles de la imagen que no contengan completamente al elemento estructurante en su entorno. La erosión reduce los contornos de los objetos, se utiliza para separar objetos que están unidos por una pequeña parte de sus contornos. (Marcos, 2006)

Usando la máscara de la figura 3.

0	1	0
1	1	1
0	1	0

Figura 3 Mascara 3x3

Y una figura A formada por la matriz

0	0	0	0	0	0	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	0	0	0	0	0	0

Figura 4 Matriz sin erosionar

Realizando el proceso de erosión la matriz queda de la siguiente forma

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Figura 5 Matriz erosionada

Dilatación binaria

Dada una máscara B (formada por unos y ceros), la dilatación de A por B es el conjunto de todos los desplazamientos de X tales que B y A se solapen en al menos un elemento distinto de cero. Por ejemplo, dada una máscara B formada por una matriz 3x3.

0	1	0
1	1	1
0	1	0

Figura 6 Mascara 3x3

Y una figura A formada por la matriz

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Figura 7 Matriz a dilatar

La matriz resultante después del proceso de dilatación se muestra en la figura 8.

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Figura 8 Matriz dilatada

Búsqueda del centroide

Método de momentos

El centroide de un objeto se obtiene calculando la media de todas las coordenadas de los puntos del contorno. La teoría de los momentos proporciona una interesante y útil alternativa para la representación de formas de objetos. Si tenemos un objeto en una región que viene dado por los puntos en los que f(x,y) > 0, definimos el momento de orden p,q como:

$$m_{pq} = \iint_{-\infty}^{\infty} x^p y^q f(x,y) dxdy$$

El interés de estos momentos generales desde el punto de vista de la caracterización discriminante de los contornos de los objetos es que, estos contornos pueden modelarse como un tipo especial de funciones f(x,y) acotadas y por ende se pueden calcular los momentos generales. En este sentido, y aquí radica su aplicabilidad en el reconocimiento de formas, dada una función acotada f(x,y) existe un conjunto de momentos generales y viceversa. Es decir, dado un conjunto de momentos generales se puede reconstruir una función f(x,y) única, simbólicamente:

$$f(x,y) \leftrightarrow \{m_{pq}\} p,q = 0,1,\ldots.\infty$$

Obviamente esta correspondencia biunívoca no representaría ningún interés práctico, a menos que se pudiera reducir el número de momentos generales a una cantidad manejable. Particularizando este enfoque basado en la descripción de una función acotada f(x,y) mediante un conjunto finito de sus momentos generales al problema del reconocimiento automático de los contornos (es decir formas cerradas) de los objetos en una imagen digital, es preciso pasar de una integral doble a un doble sumatorio, puesto que la función f(x,y) es ahora una función I(x,y) acotada que toma valores distintos de cero únicamente en el contorno del objeto y en su interior. Los momentos generales discretos de la función Io(x,y) serán pues:

$$m_{pq} = \sum_{x=0}^{N-1} \sum_{x=0}^{N-1} x^p y^q I_0(x,y) \qquad p,q = 0,1\ldots.\infty$$

Dependiendo de cómo se defina Io(x,y) se tendrá cuatro posibles momentos generales:

$$I_0(x,y) = 1\ contorno$$
$$I_0(x,y) = 0\ resto$$

$$I_0(x,y) = I(x,y)\ contorno$$
$$I_0(x,y) = 0\ resto$$

$$I_0(x,y) = 1\ objeto$$

$$I_0(x,y) = 0\ resto$$
$$I_0(x,y) = I(x,y) objeto$$
$$I_0(x,y) = 0\ resto$$

es decir se puede definir una función acotada Io(x,y) que tome los valores (1,0) o bien valores (I(x,y),0). Siendo I(x,y) el nivel de intensidad luminosa en la escala de grises. Igualmente se puede establecer el campo de existencia de la función Io(x,y) bien como el contorno del objeto o bien como el contorno mas su interior. Los momentos generales que se obtienen a partir de su contorno, aunque lo caracterizan adecuadamente, sufren el grave defecto de ser muy sensibles al ruido y a pequeñas variaciones en la forma de un contorno (siendo preciso tener un mayor número de momentos generales para robustecer la descripción del contorno). Por el contrario, los momentos generales basados en el contorno mas su interior presentan una mayor robustez. Por otro lado, para caracterizar la forma de un objeto no es necesario manejar funciones Io(x,y) multivaluadas, siendo suficiente que tomen un valor único, digamos la unidad. En consecuencia, para la obtención de características discriminantes de un contorno cerrado es habitual trabajar con la tercera opción, es decir con los momentos generales basados en la función Io(x,y):

$$I_0(x,y) = 1\ objeto$$
$$I_0(x,y) = 0\ resto$$

Un momento de gran interés es el de orden cero (el orden de un momento viene dado por la suma de los índices p y q):

$$m_{00} = \sum_{x=0}^{N-1} \sum_{x=0}^{N-1} I_0(x,y)$$

que obviamente coincide con el área del objeto. El área física se puede calcular sin mas que multiplicar m por el área física de un pixel, lo cual obliga a un calibrado previo de la cámara. Afortunadamente, desde el punto de vista del reconocimiento de objetos veremos que no es necesario pasar a las dimensiones físicas del objeto, ya que es posible obtener momentos invariantes a traslaciones, giros y tamaños relativos de los objetos. Los momentos de orden uno, m y junto con determinan el llamado centro de gravedad del objeto:

$$x = \frac{m_{10}}{m_{00}} = \frac{\sum\sum xI(x,y)}{\sum\sum I(x,y)}$$

$$y = \frac{m_{10}}{m_{00}} = \frac{\sum\sum yI(x,y)}{\sum\sum I(x,y)}$$

2.4. Microcontrolador

El Arduino Uno es una placa electrónica basada en el ATmega328. Cuenta con 14 pines digitales de entrada / salida (de los cuales 6 pueden utilizarse para salidas PWM), 6 entradas analógicas, un 16 MHz resonador cerámico, una conexión USB, un conector de alimentación, una cabecera ICSP, y un botón de reinicio. Contiene todo lo necesario para apoyar el microcontrolador; basta con conectarlo a un ordenador con un cable USB o el poder con un adaptador de CA o la batería a CC para empezar. (Arduino, 2014)

Características

Microcontrolador	ATmega328
Tensión de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límites)	6-20V
Digital pines I / O	14 (de las cuales 6 proporcionan salida PWM)
Pines de entrada analógica	6
Corriente DC por Pin I / O	40 mA
Corriente DC de 3.3V Pin	50 mA
Memoria Flash	32 KB (ATmega328) de los cuales 0,5 KB utilizado por el gestor de arranque
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Velocidad del reloj	16 MHz
Longitud	68,6 mm
Ancho	53,4 mm
Peso	25 g

2.5. Algoritmo de seguimiento Meanshift

Considere que tiene una imagen de puntos rojos, se tiene una ventana inicial, en este caso es un círculo azul, y este se tiene que mover al área donde hay mayor densidad de pixeles, como se muestra en la figura 9. (OpenCV, http://docs.opencv.org, 2014)

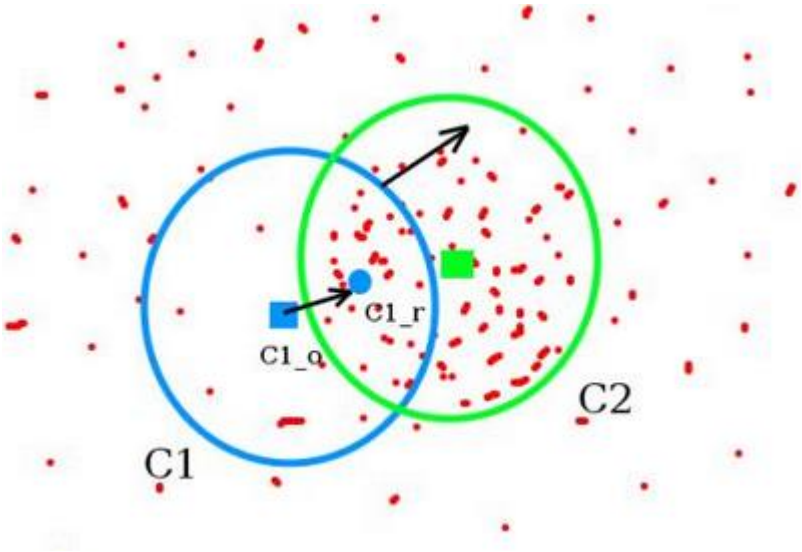


Figura 9 Proceso de iteraciones con meanshift

Iteraciones que realiza el algoritmo:

- La ventana inicial se muestra en color azul con el nombre de “C1”
- El centro de la ventana es marcado con un cuadrado azul llamado “C1_o”.
- Posteriormente se encuentra el centroide de los puntos que están dentro de la primera ventana marcado con el punto azul llamado “C1_r” el cual es el centroide real de la ventana.
- El centro del círculo se posiciona en el punto azul que corresponde al centroide en esa región.
- Nuevamente se mueve la ventana y se continúa con las iteraciones para calcular el centroide de los puntos donde se encuentra la nueva posición del círculo.

Finalmente se obtiene una ventana con la distribución máxima de pixeles, esta es marcada con el círculo verde llamado “C2”. Como se puede observar esta venta tiene el número máximo de puntos de la imagen.

A continuación se muestra el proceso en una serie de imágenes.

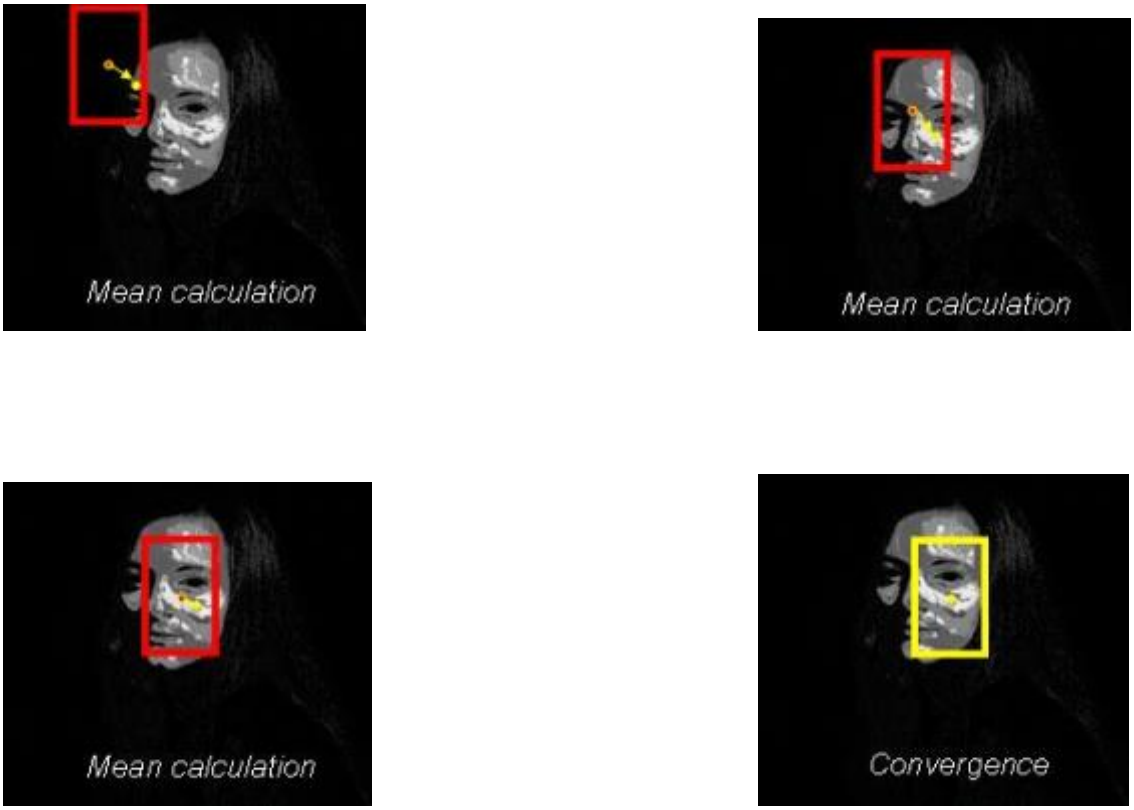


Figura 10 serie de iteraciones que realiza Meanshift tomado de (OpenCV, http://docs.opencv.org, 2014)

Capítulo 3
Estado del arte

3. Estado del arte

3.1. Trabajos Relacionados en UPIITA

Dispositivo automatizado para el control de posición de una marcadora mediante visión artificial.

Este prototipo emplea la visión artificial (que tiene como finalidad la reproducción artificial del sentido de la vista utilizando algoritmos dentro de sistemas computacionales) ya que este mecanismo controla de forma inteligente una marcadora de aire comprimido comercial automático empleada en el deporte extremo paintball, implementando el tratamiento de imágenes en un dispositivo de procesamiento el cual analiza y reconoce la presencia de un individuo obteniendo su ubicación para dirigir y realizar el accionamiento de esta hacia dicho objetivo, dentro de un área de trabajo determinada. (Jair, 2009)

Sistema de visión artificial para la navegación de un robot móvil.

El fundamento del siguiente trabajo es la adquisición de imágenes de una carretera en el que actúa un robot móvil. Estas imágenes son capturadas por medio de una cámara que la transfiere a una PC con el fin de que sean procesadas y analizadas para la toma de decisiones sobre el curso del robot.
La función de capturar, convertir y procesar la imagen tomada de la carretera, es obtener un parámetro de referencia (posición del robot con respecto a la carretera), el cual ayudara a que el robot siempre se encuentre en el centro de la misma.
El robot también es capaz de evitar obstáculos que se encuentren en la carretera, en el caso de que estos se hallen en el centro o por cuestiones de espacio no pueda evitarlos, el vehículo detendrá su marcha. (Daniel, 2009)

Sistema de visión artificial para la supervisión del control de calidad en textiles.

Este trabajo implementa un sistema inteligente que por medio de la visión artificial detecte los defectos existentes en lienzos de tela plana utilizados como materia prima en la industria textil. El objetivo primordial del sistema es diferenciar tela buena de tela defectuosa. El sistema de visión artificial se monta sobre una mesa XY, la cual permite el recorrido del lienzo de tela en dos dimensiones. Además se analizan lienzos cuadrados de tela sin color, de 80 cm por lado, posteriormente los datos son procesados por la PC. (Raúl, 2010)

3.2. Trabajos internacionales

Advanced Defenced Systems ha desarrollado un sistema militar para la intercepción de objetos aéreos en territorio israelí llamado Iron Dome. El sistema cuenta con tres elementos principales: la detección y seguimiento de objetos aéreos por medio de radar, un sistema de manejo de armas y una unidad lanzadora de misiles.

Iron Dome opera de la siguiente manera: empieza con el sistema que detecta e identifica cualquier misil lanzado hacía territorio israelí, un radar monitorea la posición del misil y con esta información el sistema de control de armas identifica la trayectoria de la amenaza y calcula de manera anticipada el punto de impacto, entonces la unidad lanzadora de misiles envía un interceptor y detona la amenaza en una zona segura, como se puede apreciar en la figura 11.

El radar que utiliza es el radar de las Fuerza Aérea Israelí, desarrollado por la empresa Elta, mientras que el sistema de control fue desarrollado por la empresa mPrest Systems y el misil utilizado para derribar los objetos cuenta con sensores optoeléctricos en los alerones. (Technology, 2012)

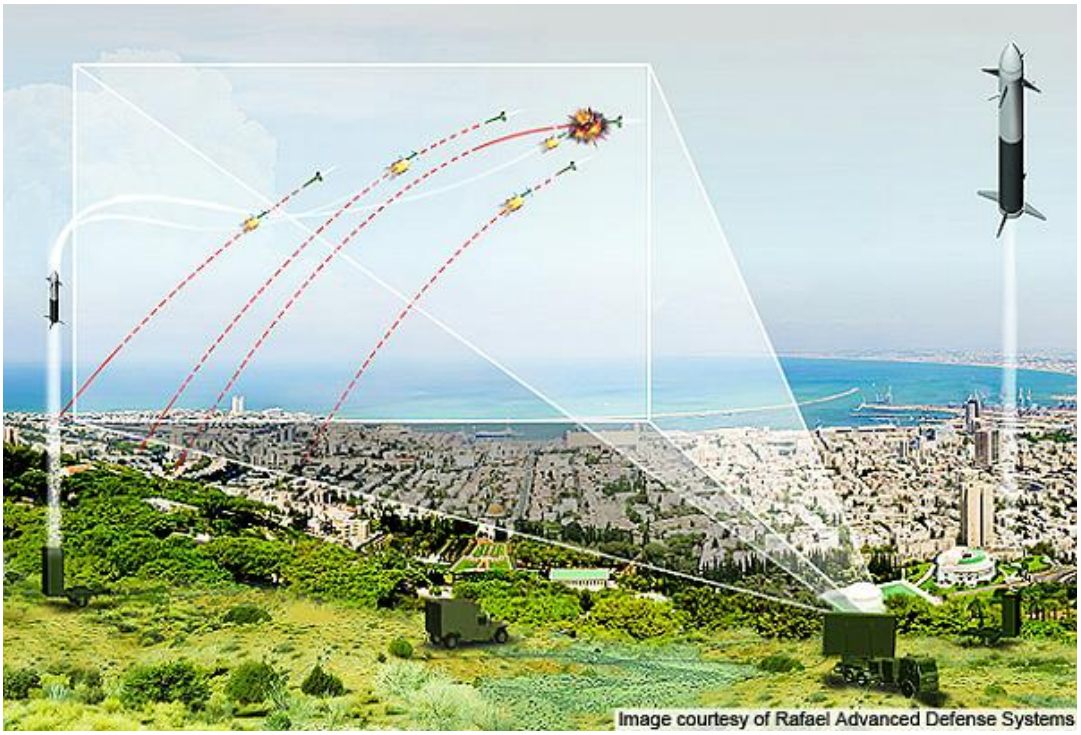


Figura 11 Funcionamiento de iron dome

También se han utilizado este tipo de sistemas en aplicaciones industriales tales como detección de errores en líneas de producción, identificación de fallas en procesos de manufactura y aplicaciones de control de calidad de productos, tal es el caso del sistema de visión compacto avanzado SBOC-M de Festo, mostrado en la Figura 12, el cual controla de manera fiable los desarrollos de movimientos rápidos en procesos de manufactura en los que los operarios no pueden percibir claramente los eventos que suceden.

El sistema cuenta con una cámara de alta velocidad y un mecanismo que le da el carácter de móvil y que le permite seguir de cerca el proceso que se esté ejecutando en la línea de producción.

Las ventajas que ofrece este sistema es la reducción en tiempo de ciclos de manufactura debido a que detecta las causas de tiempos de espera, también asegura un ajuste óptimo en la velocidad de los movimientos del proceso y el cuidado de movimientos críticos, además de la detección de vibraciones mecánicas no deseadas. (Festo, 2013)

Figura 12 Sistema de visión SBOC-M de Festo

Capítulo 4
Diseño del robot

4. Desarrollo

4.1. División por áreas funcionales

El prototipo de sistema para seguimiento de un objeto mediante visión artificial detecta un objeto previamente identificado de manera automática y mantiene una toma constante del mismo en movimiento.

El sistema de seguimiento de objetos está dividido en tres subsistemas:

- Subsistema mecánico, cuenta con tres grados de libertad que le dan la capacidad de realizar un barrido (rotación en el eje X), paneo (rotación en el eje Y) y movimiento traslacional sobre un eje.
- Subsistema de visión el cual identifica el objeto que se desea seguir mediante algoritmos de visión artificial.
- Subsistema de control, este recibe la información de la posición del objeto con referencia a la imagen previamente adquirida y realiza los ajustes adecuados de los actuadores del robot para que este no pierda el objeto.

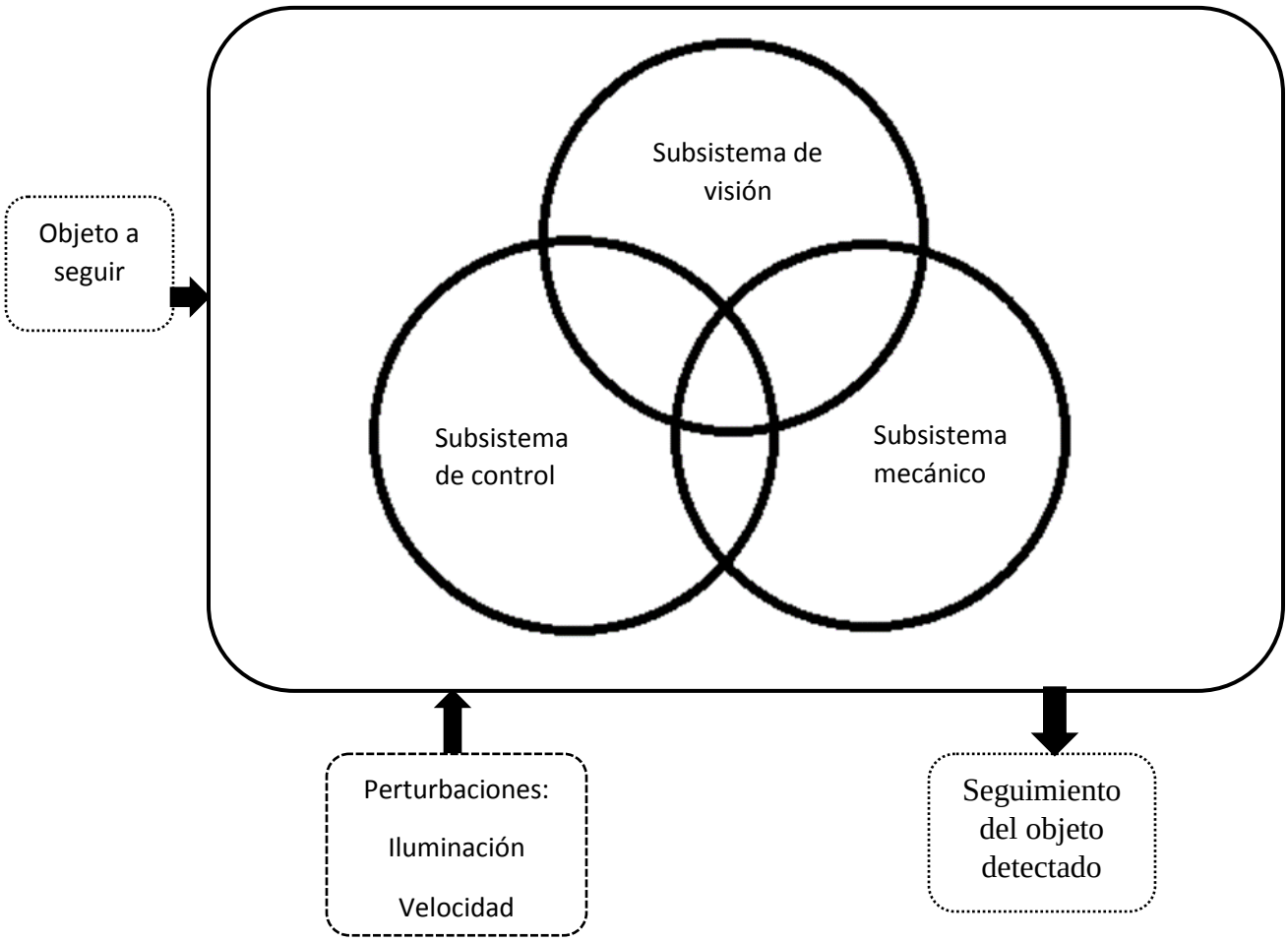


Figura 13 Diagrama de subsistemas de áreas funcionales

4.2. Diseño Detallado

4.2.1. Diseño de la estructura

El diseño de la estructura del subsistema mecánico comenzó bosquejando varias ideas en papel en búsqueda de una estructura que fuera atractiva, después de varios diseños se seleccionó la estructura que se realizó en SolidWorks obteniendo el modelo final como se muestra en la figura 14 y 15. Una vez creado el modelo se realizaron los análisis de esfuerzos correspondientes para verificar que nuestra estructura propuesta resistiera las cargas que se pretenden aplicar al robot.

Los resultados fueron satisfactorios por lo que el modelo planteado fue el definitivo.

Figura 14 Modelo de la estructura del robot

Figura 15 Diseño base de la cámara

Las dimensiones de la estructura son 219 mm X 240 mm X 140 mm, La dimensiones promedio de las cámaras de video portátiles no sobrepasan los 100 mm X 100mm, por lo que el área designada para el espacio de la cámara es de 120 mm X 120 mm.

La figura 16 muestra los componentes principales que conforman la estructura del robot.

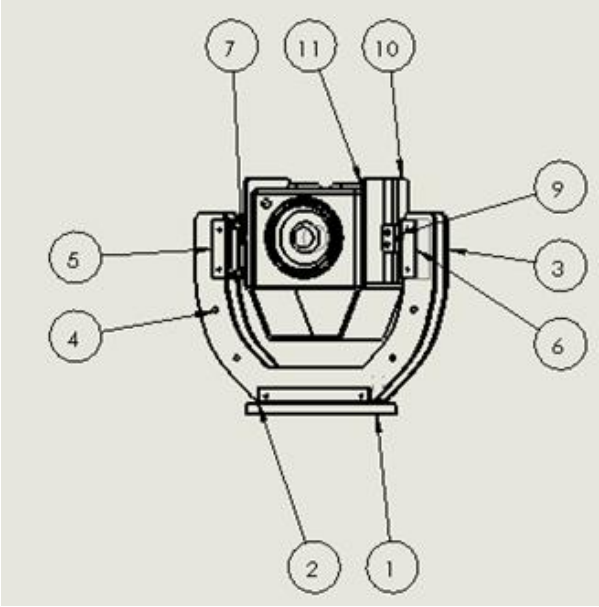


Figura 16 Lista de componentes base de la cámara

Numero de elemento	Nombre de pieza	Cantidad
1	Base circular	1
2	Separador 1	1
3	Soporte U	2
4	Perno	4
5	Separador 2	1
6	Separador del eje	1
7	Servomotor	1
8	Eje	1
9	Abrazadera	2
10	Soporte cámara	1
11	Cámara	1

Tabla 1 Lista de componentes de la estructura del robot

El mecanismo que dará movimiento lineal al robot se muestra en la figura 8, se optó por seleccionar una banda dentada debido a que una banda de sincronización no se estira ni se desliza, y en consecuencia transmite potencia a una relación constante de velocidad angular.

La velocidad transmitida entre la polea motriz y la inducida es precisa, su operación es silenciosa, son ligeras y económicas además de que tienen una gran resistencia al estiramiento.

Se buscó que el material del riel fuera fácilmente maquinable, resistente a la corrosión para uso en exteriores, de precio accesible así como ligero para su fácil manejo y transportación.

Las dimensiones del sistema de riel son de 100 mm de alto por 3000 mm de largo y 450 mm de ancho. El diseño final del subsistema mecánico se puede observar en la figura 17.

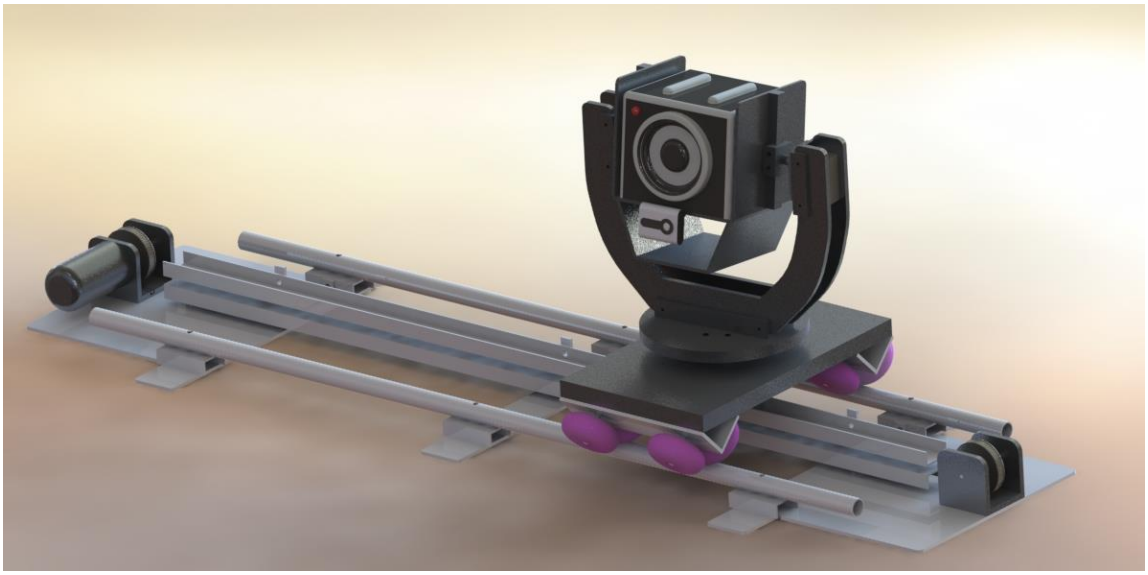
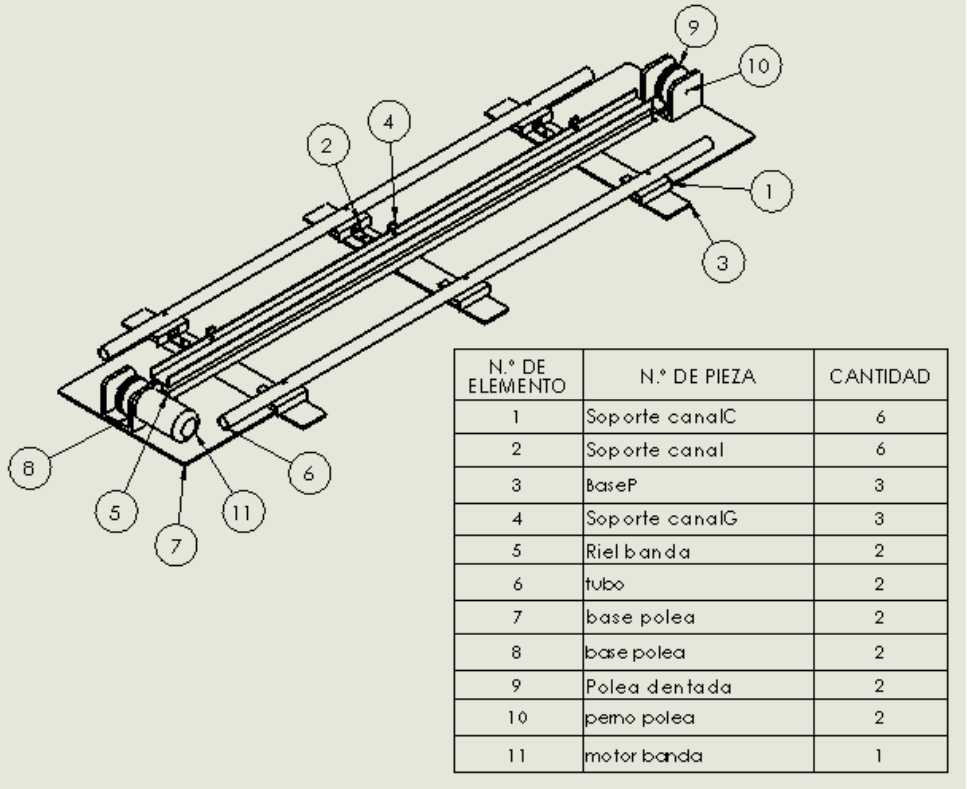


Figura 17 Renderizado del subsistema mecánico

La figura 18 muestra los principales componentes del subsistema de movimiento lineal.



N.° DE ELEMENTO	N.° DE PIEZA	CANTIDAD
1	Soporte canalC	6
2	Soporte canal	6
3	BaseP	3
4	Soporte canalG	3
5	Riel banda	2
6	tubo	2
7	base polea	2
8	base polea	2
9	Polea dentada	2
10	perno polea	2
11	motor banda	1

Figura 18 Lista de componentes del riel

El material elegido para la estructura del sistema robótico fue aluminio 6063 T5, este material se seleccionó debido a que sus propiedades mecánicas cumplen los requisitos que necesitamos, es de fácil adquisición y su costo es accesible. A continuación se muestran las propiedades mecánicas de la aleación.

Aleación en AW 6063: Aluminio-Magnesio-Silicio

Propiedades mecánicas

Estado	Espesor de la pared (mm)	Carga de rotura Rm(N/mm^2)	Limite elástico (N/mm^2)	Alargamiento (%)	Dureza Brinell (HB)
T5	e≤3	175	130	8	65
T5	3≤e≤25	160	110	7	65

Tabla 2 Propiedades mecánicas del aluminio 6063 tomado de (Díaz, 2011)

Propiedades tecnológicas:

- Resistencia a la corrosión: Muy buena
- Soldabilidad: Buena
- Conformabilidad: Buena
- Aptitud para el anodizado: Muy buena
- Maquinabilidad: Buena

Basándonos en el catálogo Master de bandas industriales de la marca Gates, nos indica que la medida máxima de la longitud de paso es de 4400mm. Por lo que es necesario optar por la banda síncrona sin fin (extremo abierto) para conseguir la longitud de paso necesaria que es de 6000mm.

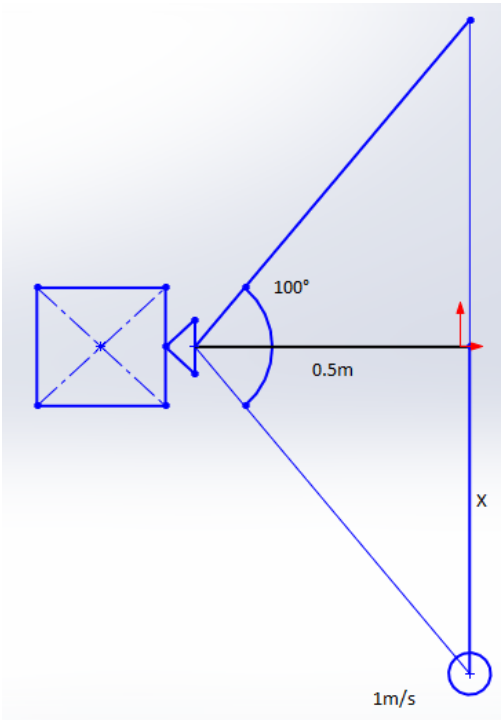
Figura 19 Banda dentada seleccionada

Características:

- Modelo: Power grip GT
- Paso: 8mm
- Ancho: 30mm
- Descripción: LL8MR30
- Diámetro de la polea: 1.25”

4.2.2. Velocidad y aceleración angular del paneo y barrido.

A continuación se muestran los cálculos realizados para obtener la velocidad y aceleración angular de la base de la cámara.



Se calculó la distancia y el tiempo que tarda un objeto con una velocidad de 1 m/s en realizar un recorrido equivalente a 50°.

$$Tan(50) = \frac{X}{0.5\ m}$$

Distancia $X = 0.5059\ m$

Tiempo $t = \frac{d}{v} = \frac{0.5059\ m}{1 m/s} = 0.5959\ \ s$

Se realizó la conversión de grados a radianes

$$\frac{(50)(\pi)}{180} = 0.87\ rad$$

Velocidad angular

$$w = \frac{0.87\ rad}{0.5959\ s} = 1.464\ \frac{rad}{s}$$

Se propone un tiempo de transitorio para llegar del reposo a una velocidad angular de 1.4 rad/s en 0.25 s y así poder calcular la aceleración angular.

Aceleración angular $\alpha = \frac{Vf - Vi}{t} = \frac{1.464\ rad/s}{0.25\ s} = 7.32\ \frac{rad}{s2}$

4.2.3. Momentos de inercia

Se calcularon los momentos de inercia de los eslabones del robot se utilizó la herramienta de SolidWorks cálculo de propiedades físicas, el cual nos da las propiedades físicas del modelo o entidades seleccionadas.
El material con el cual se realizaron los cálculos fue aluminio 6063 el cual se seleccionó en el diseño de la estructura.
Para obtener los momentos de inercia del soporte de la cámara el cual realiza el barrido de nuestro robot se tomó por separado el soporte como se muestra en la figura 20.

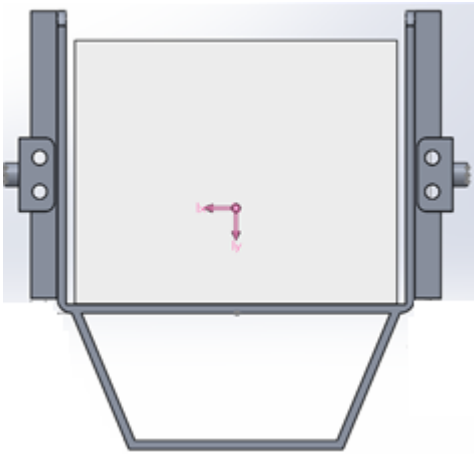


Figura 20 Ejes coordenados para el centro de masa del soporte

Se insertó un sistema de coordenadas auxiliar alineado al eje donde queremos calcular el momento de inercia como se muestra en la figura 21, nuestro eje de interés es el eje X que es el que rota y es donde se acopla el motor.

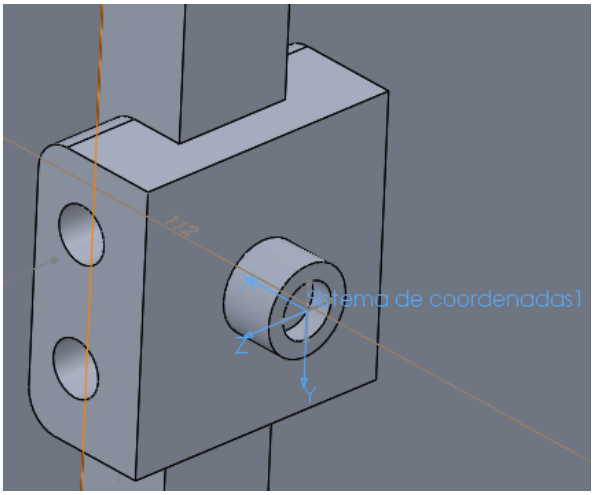
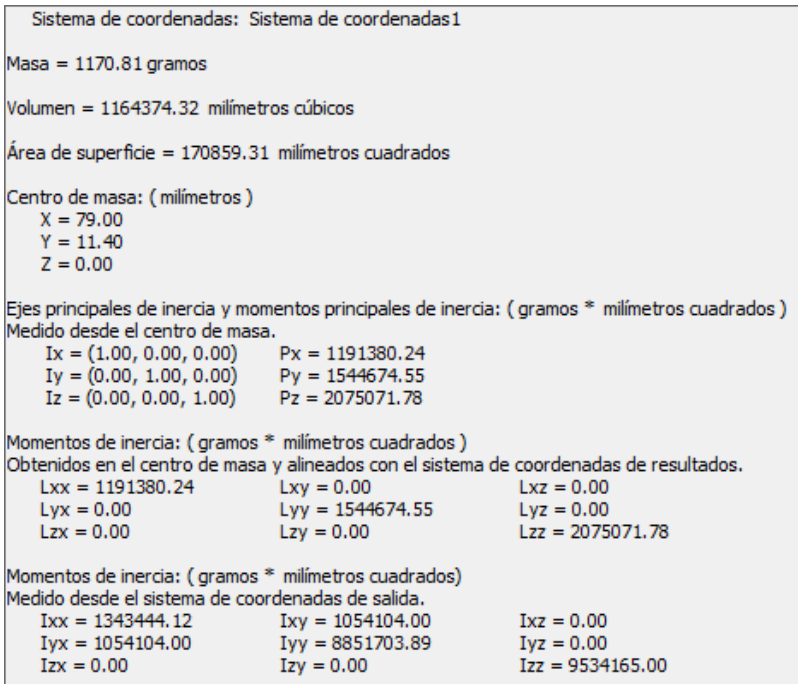


Figura 21 Sistema de coordenadas auxiliar

SolidWorks despliega las propiedades físicas del modelo de la siguiente manera, en la figura 22 se muestran tres mediciones del momento de inercia, la primera es el momento de inercia medido desde el centro de masa del modelo, la segunda es el momento de inercia obtenido en el centro de masa y alineados con el sistema de coordenadas de resultados y la tercera medición que es el momento de inercia medido desde un sistema de coordenadas de salida.



Sistema de coordenadas: Sistema de coordenadas1

Masa = 1170.81 gramos

Volumen = 1164374.32 milímetros cúbicos

Área de superficie = 170859.31 milímetros cuadrados

Centro de masa: (milímetros)
X = 79.00
Y = 11.40
Z = 0.00

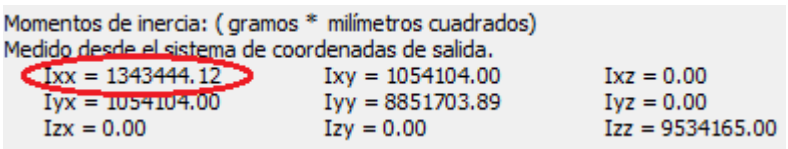
Ejes principales de inercia y momentos principales de inercia: (gramos * milímetros cuadrados)
Medido desde el centro de masa.
Ix = (1.00, 0.00, 0.00) Px = 1191380.24
Iy = (0.00, 1.00, 0.00) Py = 1544674.55
Iz = (0.00, 0.00, 1.00) Pz = 2075071.78

Momentos de inercia: (gramos * milímetros cuadrados)
Obtenidos en el centro de masa y alineados con el sistema de coordenadas de resultados.
Lxx = 1191380.24 Lxy = 0.00 Lxz = 0.00
Lyx = 0.00 Lyy = 1544674.55 Lyz = 0.00
Lzx = 0.00 Lzy = 0.00 Lzz = 2075071.78

Momentos de inercia: (gramos * milímetros cuadrados)
Medido desde el sistema de coordenadas de salida.
Ixx = 1343444.12 Ixy = 1054104.00 Ixz = 0.00
Iyx = 1054104.00 Iyy = 8851703.89 Iyz = 0.00
Izx = 0.00 Izy = 0.00 Izz = 9534165.00

Figura 22 Momentos de inercia generados en el soporte de la cámara

Para nuestro análisis se tomó la tercera medición que se muestra en la figura 23, ya que corresponde a los momentos de inercia respecto a nuestro sistema de coordenadas insertado en el punto deseado del modelo, SolidWorks utiliza el teorema de los ejes paralelos para calcular los momentos de inercia respecto a un eje propuesto.



Momentos de inercia: (gramos * milímetros cuadrados)
Medido desde el sistema de coordenadas de salida.
Ixx = 1343444.12 Ixy = 1054104.00 Ixz = 0.00
Iyx = 1054104.00 Iyy = 8851703.89 Iyz = 0.00
Izx = 0.00 Izy = 0.00 Izz = 9534165.00

Figura 23 Momentos de inercia Para el eje X del sistema de coordenadas auxiliar

Se realizó el mismo procedimiento para calcular el momento de inercia para el paneo de nuestro robot, para este caso se tomó todo el modelo completo como se muestra en la figura 24.

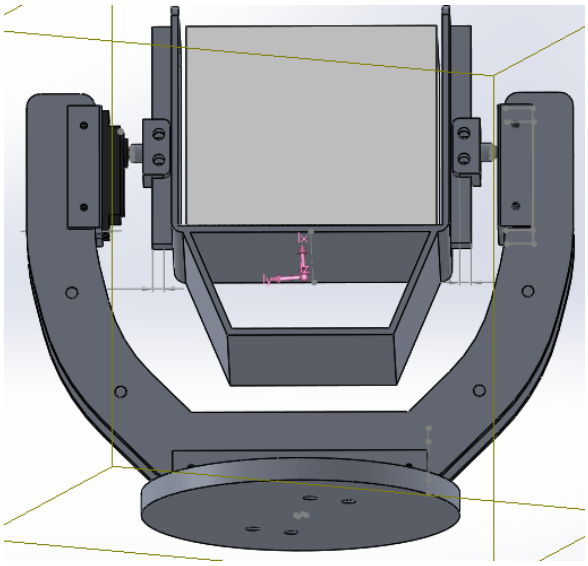


Figura 24 Ejes coordenados para el centro de masa del modelo

Se insertó un sistema de coordenadas auxiliar en la base del modelo como se muestra en la figura 25, el eje que nos interesa es el eje Y que es el que rotara para realizar el paneo del robot con ayuda del motor acoplado.

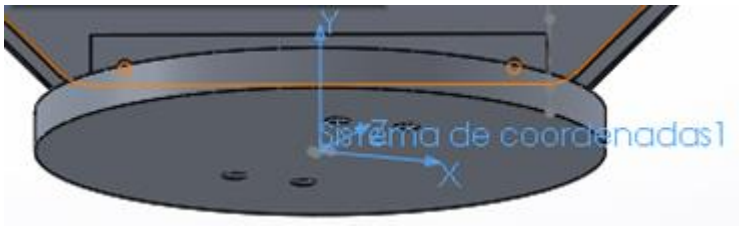


Figura 25 Acercamiento de la base con el Sistema de coordenadas auxiliar

Los resultados desplegados de las propiedades físicas del modelo se muestran a continuación.

Propiedades de masa de ensamble21-4render
Configuración: Predeterminado
Sistema de coordenadas: Sistema de coordenadas1

Masa = 2095.60 gramos

Volumen = 1523484.38 milímetros cúbicos

Área de superficie = 309829.67 milímetros cuadrados

Centro de masa: (milímetros)
X = 1.28
Y = 107.45
Z = 1.17

Ejes principales de inercia y momentos principales de inercia: (gramos * milímetros cuadrados)
Medido desde el centro de masa.
Ix = (-0.07, 1.00, -0.02) Px = 6125467.92
Iy = (-0.92, -0.07, -0.39) Py = 7199476.66
Iz = (-0.39, -0.01, 0.92) Pz = 11470418.86

Momentos de inercia: (gramos * milímetros cuadrados)
Obtenidos en el centro de masa y alineados con el sistema de coordenadas de resultados.

Lxx = 7856817.74	Lxy = -81709.39	Lxz = 1547662.76
Lyx = -81709.39	Lyy = 6131146.81	Lyz = -4497.36
Lzx = 1547662.76	Lzy = -4497.36	Lzz = 10807398.88

Momentos de inercia: (gramos * milímetros cuadrados)
Medido desde el sistema de coordenadas de salida.

Ixx = 32053623.55	Ixy = 206436.00	Ixz = 1550805.05
Iyx = 206436.00	Iyy = 6137455.80	Iyz = 259342.25
Izx = 1550805.05	Izy = 259342.25	Izz = 35004759.22

Figura 26 Momentos de inercia generados en el soporte de la cámara

El momento de inercia en el eje Y del sistema de coordenadas auxiliares de la base del modelo se muestra en la figura 27.

Momentos de inercia: (gramos * milímetros cuadrados)
Medido desde el sistema de coordenadas de salida.

Ixx = 32053623.55	Ixy = 206436.00	Ixz = 1550805.05
Iyx = 206436.00	Iyy = 6137455.80	Iyz = 259342.25
Izx = 1550805.05	Izy = 259342.25	Izz = 35004759.22

Figura 27 Momentos de inercia eje y

4.2.4. Calculo del torque para los motores encargados del paneo y el barrido.

Para el cálculo del torque se utilizó la siguiente ecuación

$$\mathrm{T} = I\ \alpha$$

Donde: $I -\ es\ el\ momento\ de\ inercia\ con\ respecto\ al\ eje\ de\ rotacion$

$\alpha -\ es\ la\ aceleracion\ angular$

La siguiente tabla muestra los valores previamente calculados de los momentos de inercia del modelo así como la aceleración angular de los eslabones.

	Barrido (eje x)	Paneo (eje y)
Momento de inercia	$0.001343\ kgm^2$	$0.006137\ kgm^2$
Aceleración angular	$7.32\ \frac{rad}{s^2}$	$7.32\ \frac{rad}{s^2}$

Tabla 3 Valores previamente calculados

Sustituyendo en la ecuación para el Barrido del robot.

$$\mathrm{T} = I_{xx}\ \alpha$$

$$\mathrm{T} = (0.001343\ kgm^2)\left(7.32\frac{rad}{s^2}\right) = 0.010248\ Nm$$

Las especificaciones comerciales de los motores manejan torques en unidades de Kg-cm por lo que es necesario hacer la conversión de nuestro torque

$$\mathrm{T} = (0.010248\ Nm)(\frac{1\ kgf}{9.81\ N})(\frac{100\ cm}{1\ m})$$= 0.1 Kg-cm

De igual forma se realizó el cálculo para el paneo del robot quedando de la siguiente manera

$$\mathrm{T} = (0.006137\ kgm^2)\left(7.32\frac{rad}{s^2}\right) = 0.044926\ Nm$$

$$\mathrm{T} = (0.044926\ Nm)(\frac{1\ kgf}{9.81\ N})(\frac{100\ cm}{1\ m})$$= 0.46 Kg-cm

4.2.5. Calculo del torque para el motor de la banda.

Para calcular el torque del motor que se encargara del movimiento lineal de la estructura que consta de la base y el soporte de la cámara, se llevó a cabo el siguiente análisis.

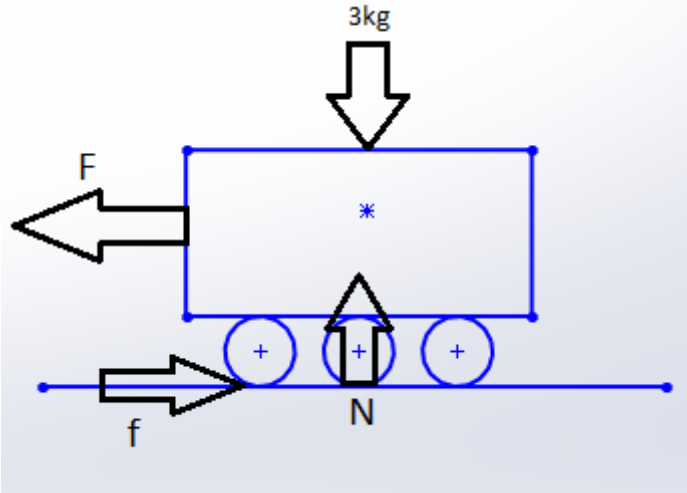


Figura 28 Diagrama de cuerpo libre del sistema robótico desplazándose

El robot tiene un peso calculado de aproximadamente 3 kg, se realizó el siguiente diagrama de cuerpo libre para observar las fuerzas que actúan sobre él y de esta manera obtener nuestra fuerza F que es la fuerza necesaria para que el móvil rompa el estado de reposo.

Donde:

F=fuerza necesaria para que el móvil comience a moverse.

N=normal del móvil.

f=fricción

Realizando sumatoria de fuerzas.

$\sum Fy = 0$ Ya que la normal y la masa de nuestro móvil se eliminan.

$\sum Fx = F = f$ La fuerza F es igual a la fricción estática que genera el móvil con el área de contacto del riel.

La fricción estática está dada por $f = (N)(\mu_e)$

Donde:

N=normal del móvil

μ_e=coeficiente de fricción (μ_e para aluminio contra plástico es de 0.2)

Sustituyendo la formula $F = f = (3\ kg)\left(9.81\ \frac{m}{s^2}\right)(0.2) = 5.886\ N$

El valor de la fuerza se utilizó para calcular el torque mínimo para poder mover el sistema robótico de forma lineal.

$$T = (F)(R)$$

Donde:

F=fuerza

R=radio de la polea

T=(5.886 N)(0.015 m)=0.093 Nm=0.94 Kgcm

4.2.6. Rodamientos de polea inducida.

Basándonos en el catálogo Rodamientos FAG, se seleccionaron los siguientes rodamientos rígidos de bolas.

Características:

- Diámetro interior: 6mm
- Diámetro exterior: 19mm
- Capacidad de carga dinamica:2.55kN
- Capacidad de carga estática: 1.04kN
- Descripción: 626FAG

Figura 29 Rodamientos seleccionados para la polea inducida

Capítulo 5
Subsistema de visión artificial

5. Sistema de visión artificial

5.1. Hardware y software

A continuación se mencionan las características del hardware y software utilizado para el desarrollo del sistema de visión artificial.

5.1.1. Hardware

El hardware que tiene como finalidad capturar las imágenes digitales del objeto a seguir, se detallan a continuación:

Webcam: El sensor de visión que se utiliza para la obtención de imágenes del objeto a seguir, es una webcam ViBook, cuenta con un sensor CMOS (*complementary metal oxide semiconductor*), tiene una resolución VGA (640X480) con una velocidad de transmisión de 30 cuadros por segundo y un ángulo visual de 30°.

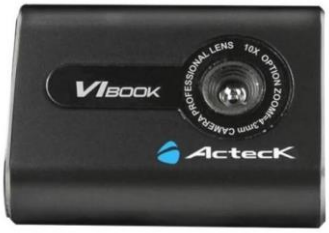


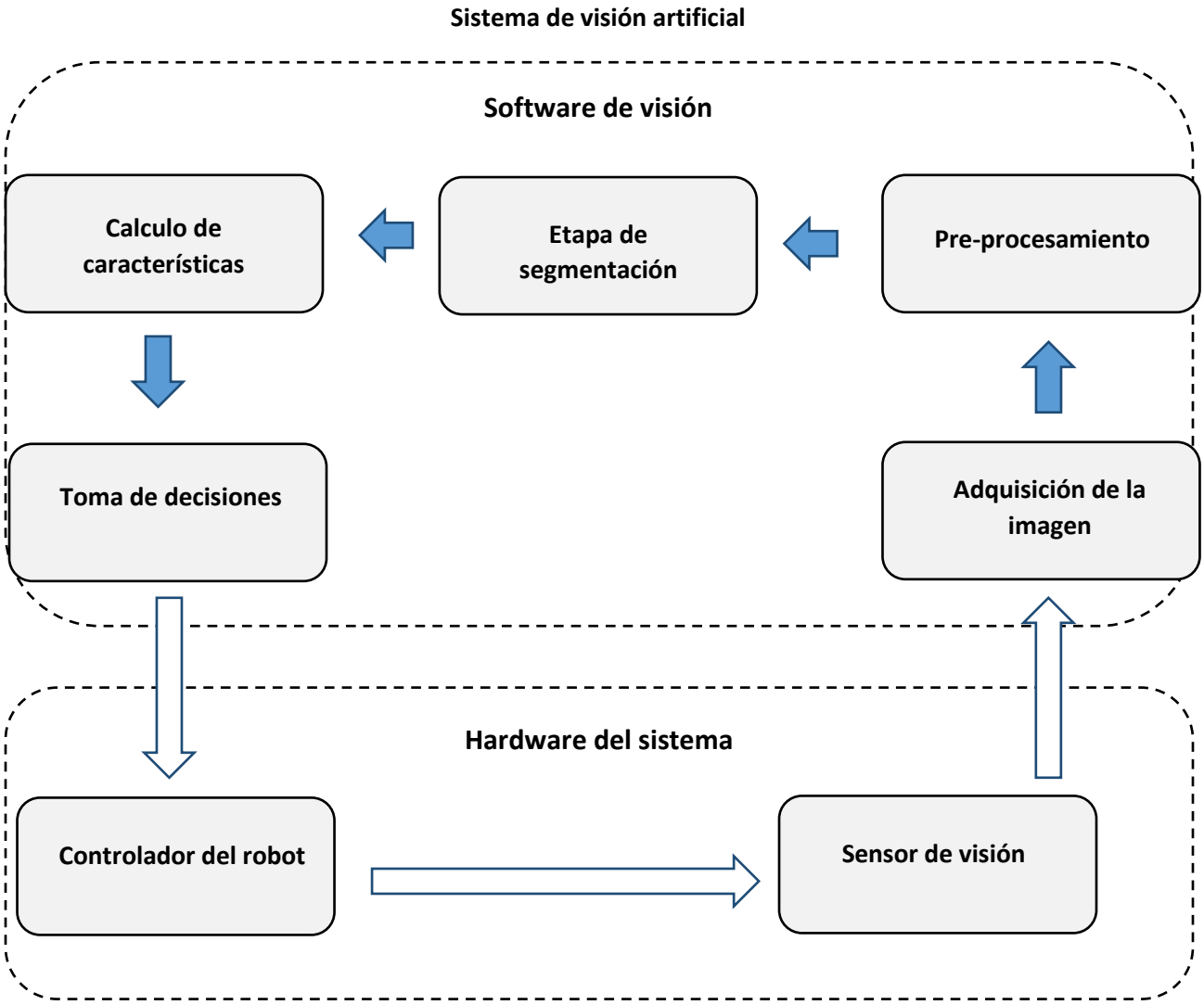
Figura 30 Webcam Vibook

5.1.2. Software

El software de visión artificial para seguimiento de objetos fue desarrollado en lenguaje de programación Visual Basic C++ con la ayuda de las funciones que nos ofrece la librería OpenCV. Sin embargo para validar el desarrollo de los algoritmos implementados en dicho lenguaje, usamos previamente Matlab, debido a las facilidades que este lenguaje ofrece para la lectura y procesamiento de imágenes.

5.2. Diagrama a bloques del sistema de visión

El objetivo del sistema de visión para el robot, es el de transformar la imagen del medio ambiente, proporcionada por la cámara, en una descripción de la posición actual del objeto a seguir. Dicha descripción deberá contener información necesaria para que el robot efectúe los movimientos que permitirán la ejecución de la tarea programada. Para alcanzar estos objetivos, se elaborar las siguientes funciones:



5.3. Preprocesamiento

La utilización de estas técnicas permite el mejoramiento de las imágenes digitales adquiridas de acuerdo a los objetivos planteados en el sistema de visión. A continuación se mencionan las técnicas de preprocesamiento empleado en el presente trabajo.

5.3.1. Convertir imagen del espacio RGB a HSV

La imagen obtenida por el sensor de visión se encuentra en el modelo de color RGB, para realizar un mejor filtrado de los colores del objeto a seguir se transforma el modelo de color a HSV utilizando la siguiente función.

Función de OpenCV: cvtColor(CV_BGR2HSV**)**

La función convierte una imagen de entrada de un espacio de color a otro.

C ++: **void cvtColor** (InputArray **src** , outputArray **dst** , int **código** , int **dstCn** = 0)

Parámetros:

- **src -** imagen de entrada: 8 bits sin signo.
- **código -** código de color conversión del espacio.
- **dstCn -** número de canales de la imagen de destino; si el parámetro es 0, el número de los canales se deriva automáticamente de src.

Valor de salida:

dst - imagen de salida del mismo tamaño y profundidad como src .

5.3.2. Umbralización para filtrar el color deseado.

La umbralización elimina los valores superiores o inferiores respecto a un valor conocido como umbral. Estos valores si ingresan manualmente al programa para detectar el color exacto del objeto a seguir.

La función utilizada dentro del algoritmo es la siguiente:

Función de OpenCV: inRange( )

Comprueba si el elemento de la matriz se encuentra entre los valores de umbral mínimo y máximo.

C ++: **void INRANGE** (InputArray **src** , InputArray **lowerb** , InputArray **upperb** , outputArray **dst**)

Parámetros de entrada:

src - primera matriz de entrada.

lowerb - límite inferior.

upperb - límite superior.

Valor de salida:

dst - matriz de salida del mismo tamaño que src.

La función comprueba el rango de la siguiente manera:

Por cada elemento de una matriz de entrada de un solo canal:

$$\texttt{dst}(\texttt{I}) = \texttt{lowerb}(\texttt{I})_0 \leq \texttt{src}(\texttt{I})_0 \leq \texttt{upperb}(\texttt{I})_0$$

5.4. Etapa de segmentación - Operaciones morfológicas

Obtenida la imagen umbralizada del objeto a seguir se procede a realizar las operaciones morfológicas.

Se eliminan los pixeles no deseados o ruido que pueda tener nuestra imagen binarizada mediante una erosión de la imagen, la función utilizada se muestra a continuación.

Función de OpenCV: Erode( )

Erosiona una imagen mediante el uso de un elemento estructurante específico.

C ++: **void erode** (InputArray **src** , outputArray **dst** , InputArray **kernel** ,

Punto de anclaje = punto (-1, -1), int **iteraciones** = 1)

Parámetros de entrada:

src - imagen de entrada; el número de canales puede ser arbitraria, pero la profundidad debe ser uno deCV_8U.

Kernel - elemento estructurante utilizado para la erosión.

Ancla - posición del anclaje dentro del elemento; valor predeterminado (-1, -1) significa que el ancla está en el centro del elemento.

Iteraciones - número de veces que la erosión se aplica a la imagen.

Valor de salida:

dst - imagen de salida del mismo tamaño y tipo que src .

La función erosiona la imagen utilizando el elemento estructurante especificado que determina la forma de una vecindad del píxel sobre el que se toma el mínimo:

$$\texttt{dst}(x,y) = \min_{(x',y'):\, \texttt{element}(x',y')\neq 0} \texttt{src}(x+x', y+y')$$

La erosión puede ser aplicada varias veces. En el caso de imágenes de múltiples canales, cada canal se procesa independientemente.

Posteriormente se realiza una dilatación de la imagen para recuperar la información perdida de nuestro objeto a seguir durante la erosión, la función utilizada en nuestro algoritmo fue la siguiente.

Función de Open CV: Dilate ( )

Dilata una imagen mediante el uso de un elemento estructurante específico.

C ++: **void dilate** (InputArray **src** , outputArray **dst** , InputArray **kernel** ,

Punto de anclaje = punto (-1, -1), int **iteraciones** = 1)

Parámetros de entrada:

src - imagen de entrada; el número de canales puede ser arbitraria, pero la profundidad debe ser uno deCV_8U , CV_16U , CV_16S , CV_32F.

Kernel - elemento estructurante utilizado para la dilatación.

Ancla - posición del anclaje dentro del elemento; valor predeterminado (-1, -1) significa que el ancla está en el centro del elemento.

Iteraciones - número de veces que se aplica la dilatación.

Valor de salida:

dst - imagen de salida del mismo tamaño y tipo que src .

5.5. Calculo de características – Búsqueda de contornos

Los bordes se encuentran en zonas de una imagen donde el nivel de intensidad cambia bruscamente, cuanto más rápido se produce el cambio de intensidad el eje o borde es más fuerte. En general los bordes en una imagen se pueden distinguir por los cambios de valor de entre 2 o más pixeles adyacentes. Podemos clasificar los bordes según sea su dirección, bordes verticales, cuando los pixeles conectados verticalmente tienen valores diferentes respecto al anterior o anteriores, bordes horizontales, cuando se tienen pixeles conectados horizontalmente y estas tienen distintos valores respecto al anterior y posterior.

La función utilizada para la detección de bordes se muestra a continuación.

Función de OpenCV: findContours( )

Encuentra contornos en una imagen binaria.

C++: **void findContours**(InputOutputArray **image**, OutputArrayOfArrays **contours**, OutputArray **hierarchy**, int **mode**, int **method**)

Parámetros:

image - Imagen de un solo canal de 8 bits. Los píxeles no-cero son tratados como 1. Los píxeles con valor cero permanecen 0, por lo que la imagen se trata como binaria. La función modifica la imagen, mientras extrae los contornos.

hierarchy - vector de salida opcional, contiene información sobre la topología de la imagen. Se tiene tantos elementos como el número de contornos.

mode -Modo de recuperación del contorno.

CV_RETR_CCOMP recupera todos los contornos y los organiza en una jerarquía de dos niveles. En el nivel superior, se almacenan los contornos exteriores de los componentes, en el segundo nivel se encuentran los contornos de los agujeros.

method - Método de aproximación de contorno

CV_CHAIN_APPROX_SIMPLE comprime segmentos horizontales, verticales y diagonales y deja sólo sus puntos finales. Por ejemplo, un contorno rectangular se codifica con 4 puntos.

Valor de salida:

contours – Contornos detectados. Cada contorno se almacena como un vector de puntos.

La función recupera los contornos de la imagen binaria utilizando el siguiente algoritmo.

Análisis estructural topológico de imágenes binarias por frontera siguiente.

El método de la frontera siguiente es una de las técnicas fundamentales en el procesamiento de imágenes binarizadas, este deriva una secuencia o cadena generada por las coordenadas entre una componente de valor 1 que se encuentra conectada a una componente de valor 0, esta técnica se ha estudiado profundamente debido a que tiene una gran variedad de aplicaciones como lo es comprensión de imágenes, reconocimiento y análisis de imágenes. (Suzuki, 1985)

El algoritmo funciona de la siguiente manera, comienza escaneando la imagen de entrada binarizada, se interrumpe el rastreo cuando un pixel (i,j) satisface la condición de punto de inicio de borde como muestra la fig.31a o si encuentra un borde central como muestra la fig.31b.

Si el pixel (i,j) satisface ambas condiciones, (i,j) se considera como un punto inicial de un borde exterior.

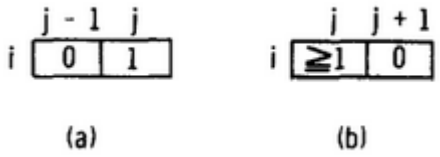


Figura 31 a-b

Cuando se encuentra el punto de inicio del borde, se comienza a preguntar por los pixeles vecinos en dirección horaria, cuando se encuentra un pixel que corresponde al borde se brinca a ese pixel y se pregunta de nuevo en búsqueda de un pixel de contorno, de tal manera que solo se hace el barrido del borde de la imagen, por lo que no se tiene que realizar un barrido de toda la imagen reduciendo el tiempo de procesamiento de la imagen.

Descripción del algoritmo

Dada una imagen de entrada $F = (f_{ij})$, se inicia la variable NBD a 1, que será el borde de toda la imagen, se escanea la imagen y se realizan los siguientes pasos a cada pixel que sea diferente de cero $f_{ij} \neq 0$.

(1) Selecciona una opción:
   (a) Si $f_{ij} = 1$ y $f_{i,j-1} = 0$ se decide si el pixel (i,j) es el punto de inicio de un borde exterior, y se incrementa NBD,($i_2, j_2) \leftarrow (i, j-1)$.

   (b) De lo contrario si $f_{ij} \geq 1\ y\ f_{i,j+1} = 0$ entonces se decide si el pixel (i,j) es el punto de inicio de un borde de agujero, se incrementa NBD,$(i_2, j_2) \leftarrow (i, j+1)\ and\ LNBD \leftarrow f_{ij}\ en\ caso\ f_{ij} > 1$.

   (c) De cualquier otra forma ve a (3).

(2) Desde el punto de inicio del borde exterior (i,j), se sigue el borde detectado mediante los siguientes pasos.

   (2.1) comenzando del punto (i_2, j_2) se realiza un barrido en el sentido de las manecillas del reloj sobre los pixeles vecinos de (i,j) hasta encontrar un pixel no cero. Suponiendo que el pixel (i_1, j_1) es el primer pixel no cero encontrado.
   Si no se encuentra un pixel no cero, se asigna –NBD a f_{ij} y se regresa a (3).

   (2.2) $(i_2, j_2) \leftarrow (i_1, j_1)\ y\ (i_3, j_3) \leftarrow (i_1, j_1)$.

   (2.3) comenzando del siguiente elemento del pixel (i_2, j_2) en el sentido de las manecillas del reloj, se examinan los vecinos del pixel (i_3, j_3) para encontrar un pixel no cero, que será (i_4, j_4).
   (2.4) se cambia el valor $(f_{i3,j3})$ del pixel (i_3, j_3) como se muestra:

      (a) si el pixel $(i_3, j_3 + 1)$ es un pixel cero examinado en el paso (2.3), entonces $f_{i3,j3} \leftarrow -NBD$.

      (b) si el pixel $(i_3, j_3 + 1)$ no es un pixel cero examinado en el paso (2.3) y $f_{i3,j3} = 1$, entonces $f_{i3,j3} \leftarrow NBD$.
      (c) de cualquier otra forma, no se cambia $f_{i3,j3}$

   (2.5) si $(i_4, j_4) = (i, j)\ y\ (i_3, j_3) = (i_1, j_1)$(volviendo al punto de partida), entonces ve a (3), de cualquier otra forma, $(i_2, j_2) \leftarrow (i_3, j_3), (i_3, j_3) \leftarrow (i_4, j_4)$, y regresa al punto (2.3)

   (3) Si $f_{ij} \neq 1$, entonces LNBD$\leftarrow (f_{ij})$ y se reanuda el rastreo desde el pixel (i,j+1).

El algoritmo termina cuando el escaneo alcanza la esquina inferior derecha de la imagen. (Suzuki, 1985)

5.5.1. Búsqueda del centroide

El centroide del objeto se obtiene calculando la media de todas las coordenadas de los puntos del contorno, la siguiente función nos entrega los momentos del objeto a seguir detectado.

Función de OpenCV: Moments( )

Calcula todos los momentos hasta el tercer orden de un polígono o forma rasterizada.

C ++: moments (InputArray **array** , bool **binaryImage** = false)

Parámetros:

array - Imagen de trama (un solo canal, de 8 bits o de punto flotante 2D matriz).

binaryImage - Si es true, todos los pixeles no cero de la imagen son tratados como 1.

Valor de salida:

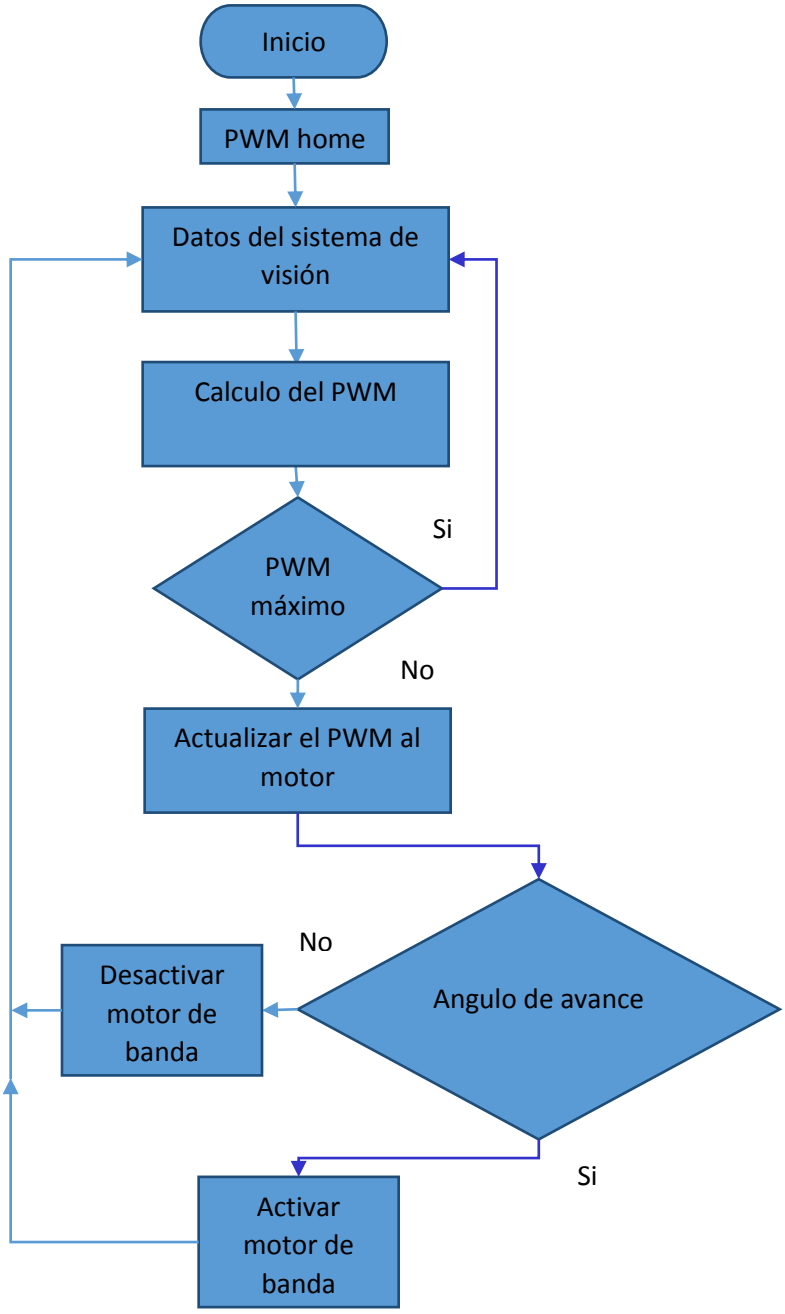
moments - momentos de salida.

Realizadas estas técnicas de procesamiento de la imagen se obtiene la información necesaria para enviarla al subsistema de control y poder realizar los movimientos pertinentes para que el seguimiento del objeto detectado se realice.

Capítulo 6
Control del sistema

6. Control del sistema – Toma de decisiones

El siguiente diagrama de flujo explica el control del sistema de seguimiento de objetos mediante visión artificial, como se interpreta la información entregada por el sistema de visión y como se implementan las condicionales necesarias dentro del micro controlador para poder controlar los actuadores del robot.



El robot comienza en una posición inicial de paneo y barrido con un valor X del PWM que controla los servomotores, el robot tiene un ángulo de apertura de aproximadamente 220°.

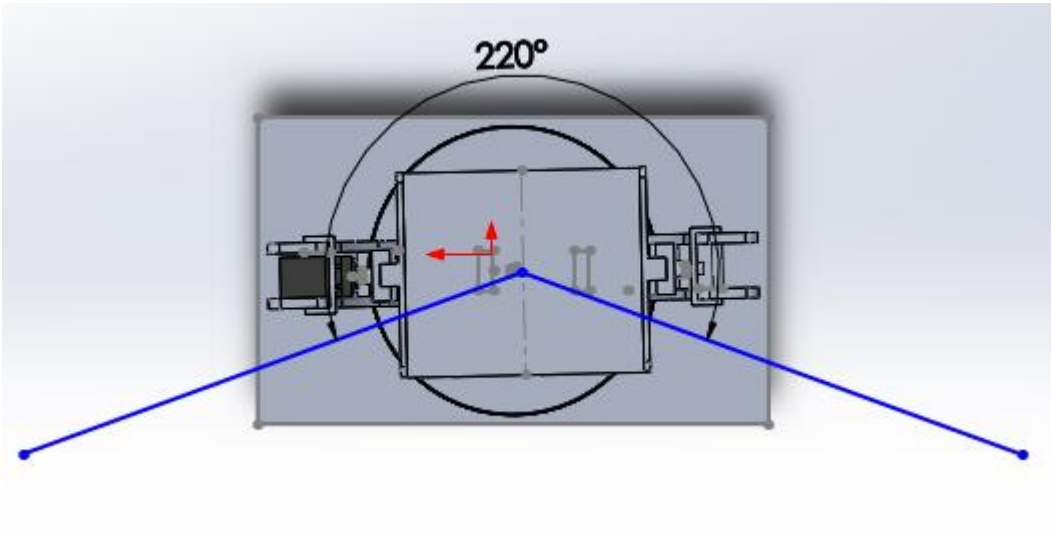


Figura 32 Apertura del robot

La etapa de cálculo de características del sistema de visión artificial entrega las coordenadas "X" y "Y" del objeto detectado, el objeto siempre debe situarse lo más cerca del centro de la imagen, por lo que con estos valores se calcula el error del objeto respecto al centro de la imagen que se está adquiriendo constantemente.

El cálculo del error tanto en "X" como en "Y" nos da la magnitud del cambio que debe tomar la señal del PWM respectiva de cada servomotor, por lo tanto también nos indica la dirección hacia donde se deberán mover los servomotores y la velocidad con la que cambian de posición.

La imagen tiene una altura de 480 pixeles por 640 pixeles de ancho, el valor máximo que puede llegar a tomar el PWM del eje "Y" es 140 y el mínimo de 90, para el PWM del eje "X" se toma un valor mínimo de 80 y un máximo de 130, este valor se limita debido a que el servomotor puede girar más de 360° provocando que el potenciómetro que cierra el lazo de control de posición se salga de escala y comience a variar la posición de manera errónea.

El control implementado para el cálculo de la señal PWM equivale a un control proporcional, la salida del controlador es proporcional a la señal de error, que es la diferencia entre el punto objetivo que se desea y la variable de proceso.

La fórmula general para obtener el PWM es:

$$PWM = PWM_{actual} \pm \frac{abs(error)}{(No.\,de\ pixeles\ del\ eje)(2)}$$

Donde la ganancia proporcional para el eje Y es

$$\frac{1}{(240)(2)}$$

Y la ganancia proporcional para el eje X es

$$\frac{1}{(320)(2)}$$

La figura 33 muestra como está dividida el área de la imagen para calcular el error respecto al eje Y.

ErrorY= 240 – coordenada Y del centroide

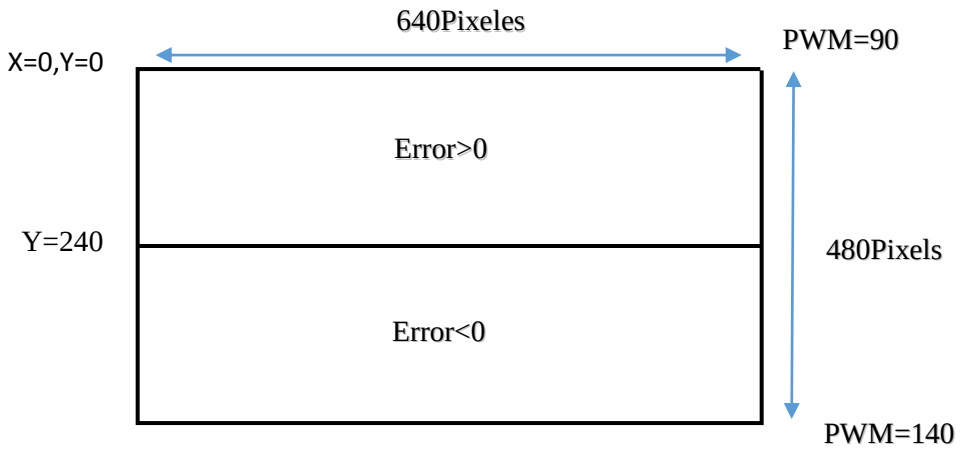


Figura 33 Error en eje Y

Donde el error puede tomar valores positivos o negativos, si el error es positivo (Error>0) y el valor actual del PWM es menos a 140, el valor del PWM se calcula con la siguiente ecuación.

$$PWM_Y = PWM_Y - \frac{abs(error_Y)}{(240)(2)}$$

Si el error es negativo (Error<0) y el valor actual del PWM es mayor a 90 la ecuación que se utiliza será.

$$PWM_Y = PWM_Y + \frac{abs(error_Y)}{(240)(2)}$$

Para el cálculo del error en el eje X se realiza el mismo método

ErrorX=320-Coordenada X del centroide

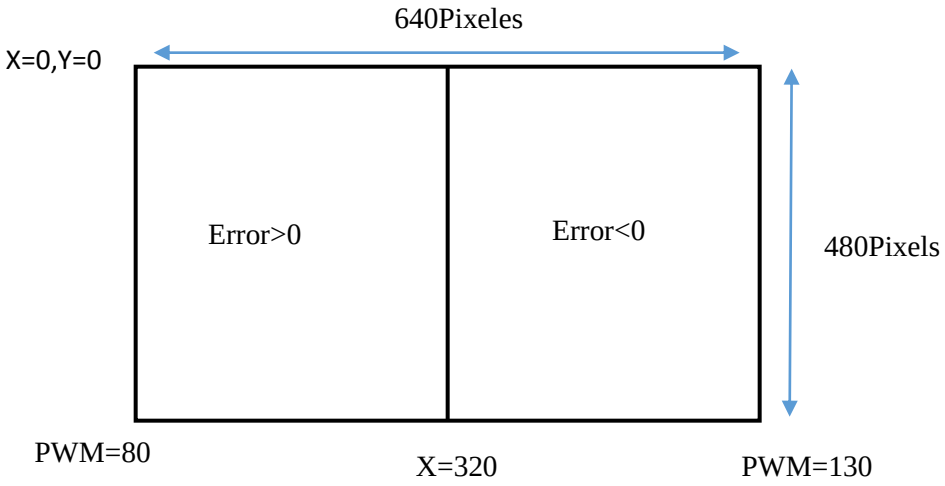


Figura 34 Error en eje X

Donde el error puede tomar valores positivos o negativos, si el error es positivo (Error>0) el valor del PWM se calcula con la siguiente ecuación.

$$PWM_X = PWM_X - \frac{abs(error_X)}{(320)(2)}$$

Si el error es negativo (Error<0) la ecuación que se utiliza será.

$$PWM_X = PWM_X + \frac{abs(error_X)}{(320)(2)}$$

La posición del objeto es monitoreada constantemente, por lo que se realizan los ajustes necesarios a las señales del PWM de forma inmediata, dando así la retroalimentación necesaria para que el sistema pueda mantenerse siempre enfocando al objeto.

A continuación se explican las condicionales necesarias para el control del movimiento lineal. Para controlar el movimiento lineal del robot se monitorea el rango en el que se encuentra el PWM, lo que nos indica el ángulo al que el robot se encuentra.

Se implementó un controlador todo o nada multiposición el cual contiene una combinación de histéresis con la zona muerta como se muestra en la figura 35.

- **Zona Muerta (H1=2)** el robot se mantendrá en estado de reposo
- **Zona activa (H1=1)** el robot se desplaza hacia la derecha
- **Zona activa (H1=3)** el robot se desplaza hacia la izquierda
- **Zona histéresis:** El robot se mantendrá realizando la acción de la última zona donde se localizó antes de entrar a la región de histéresis.

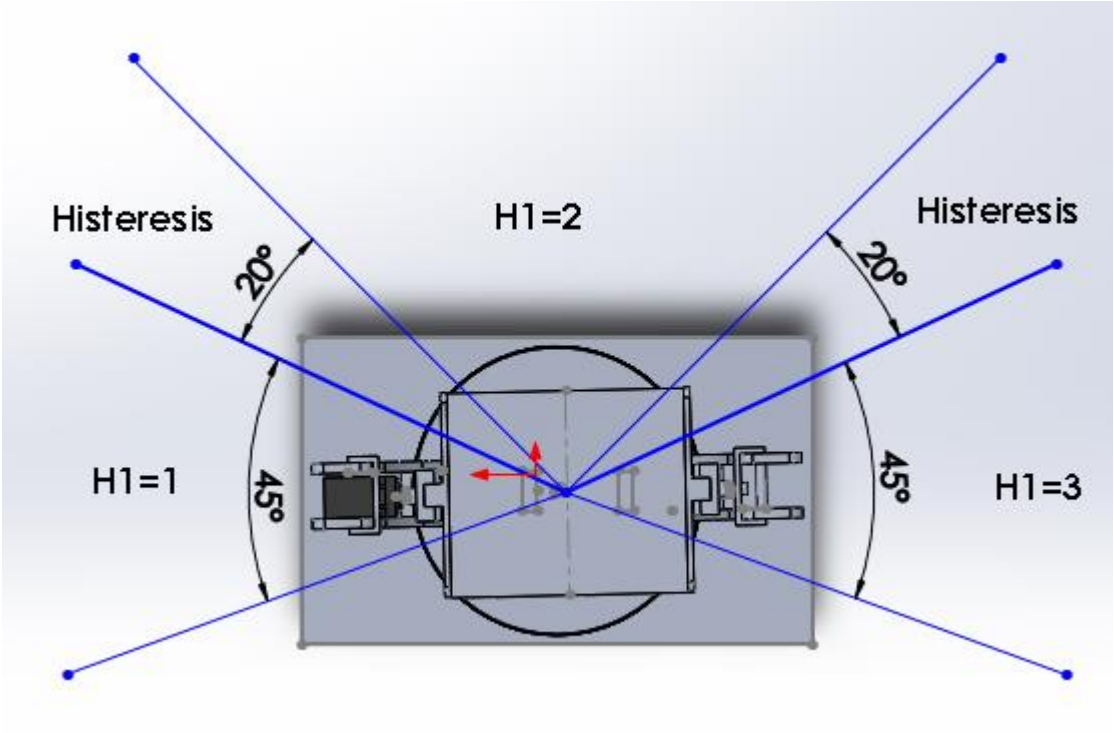


Figura 35 Zonas del controlador multiposición

Mediante los ciclos de la histéresis se evitan las conmutaciones bruscas entre el giro en uno de los dos sentidos y el paro. Mediante la zona muerta se evitan las conmutaciones bruscas entre un sentido de giro y otro.

Se utilizan sensores inductivos de fin de carrera para indicar que el robot ha llegado al límite del riel.

6.1. Dispositivos del subsistema de control

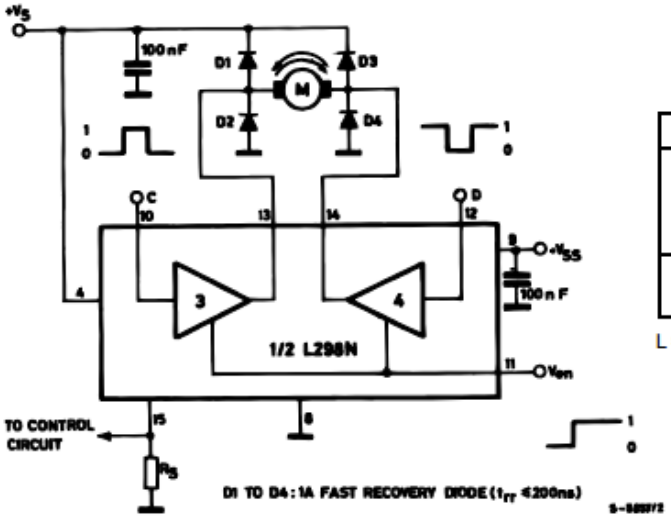
6.1.1. Driver para motor de corriente directa

El driver utilizado para la inversión del giro del motor de CD el cual es el encargado del movimiento lineal del robot es el L298N.

El módulo permite controlar el sentido de giro y velocidad mediante señales TTL que se obtienen del microcontrolador, cuenta con todos los componentes necesarios para funcionar sin necesidad de elementos adicionales, entre ellos diodos de protección y un regulador LM7805 que suministra 5V a la parte lógica del integrado L298N.

Cuenta con jumpers de selección para habilitar cada una de las salidas del módulo (A y B). La salida A está conformada por OUT1 y OUT2 y la salida B por OUT3 y OUT4. Los pines de habilitación son ENA y ENB respectivamente.

El diagrama eléctrico del circuito es el sugerido en la hoja de datos del componente, control bidireccional de un motor de CD.



Inputs		Function
V_{en} = H	C = H ; D = L	Forward
	C = L ; D = H	Reverse
	C = D	Fast Motor Stop
V_{en} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't care

Figura 36 Diagrama eléctrico tomado de (STMicroelectronics, 2000)

6.1.2. Características del driver L298N

- La tarjeta tiene la opción de habilitar un regulador LM7805 integrado en ella para alimentar la parte lógica con lo que se puede alimentar la tarjeta con 12V por ejemplo.
- Corriente máxima 4 Amperios.
- Posee 6 entradas de control
- Admite entradas de señal PWM para el control de velocidad.

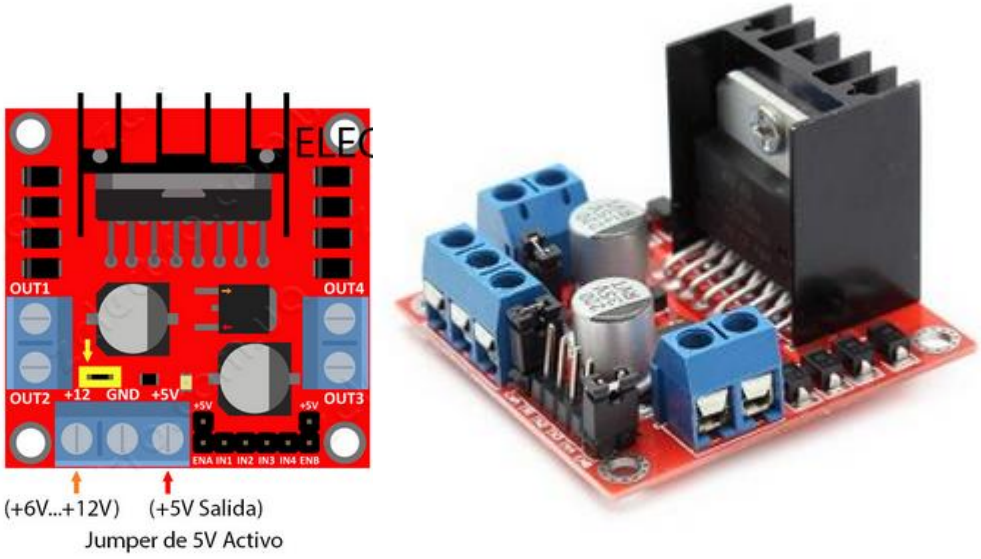


Figura 37 Diagrama físico del L298N

6.1.3. Comunicación del sistema de visión con el sistema de control

La comunicación del sistema de visión artificial (PC) con el sistema de control (Microcontrolador) se realizó utilizando el protocolo de comunicación serial. La comunicación serial es un protocolo muy común para comunicación entre dispositivos que se incluye de manera estándar en prácticamente cualquier computadora. La comunicación serial es también un protocolo común utilizado por varios dispositivos para instrumentación, es por eso que tanto Arduino como Visual Studio contienen funciones para habilitar una transmisión de datos mediante comunicación serial.

La mayoría de los microcontroladores, entre ellos Arduino, poseen un puerto de comunicación serial. Para comunicarse con los computadores personales actuales que poseen únicamente puerto USB requieren de un dispositivo "traductor". Arduino emplea el integrado FT232R, el cual es un convertidor USB-Serial. A través de este integrado el microcontrolador puede recibir y enviar datos a una computadora de manera serial.

La parte física encargada de la comunicación serial es la UART (Universal Asynchronous Receiver and Transmitter). Los microcontroladores Atmega8/168/328, en los cuales está basado Arduino, disponen de un dispositivo compatible llamado USART (Universal Synchronous and Asynchronous serial Receiver and Transmitter) que permite tanto la comunicación asincrónica como sincrónica.

6.1.4. Enviando datos al arduino

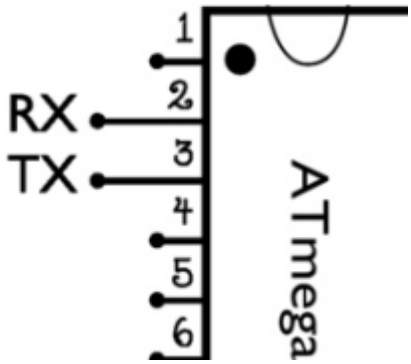


Figura 38 Pines de transmisión serial tomado de (Atmel, 2009)

Pines encargados de la comunicación serial en los microcontroladores Atmega8, Atmega 168 y Atmega328

En la comunicación con la computadora Arduino emplea la comunicación asincrónica. Esto es, requiere de sólo dos líneas de conexión que corresponden con los pines 2 y 3: Pin 2 (Rx) pin de recepción y pin 3 (Tx) pin de transmisión, y del establecimiento de un nivel de tierra común con la computadora, esto es, ambas tierras deben estar conectadas, estableciendo el mismo nivel de voltaje de referencia.

Además de realizar las conexiones físicas entre el microcontrolador y la computadora, para que pueda establecerse la comunicación serial debe existir un acuerdo previo en la manera cómo van a ser enviados los datos. Este acuerdo debe incluir los niveles de voltaje que serán usados, el tamaño y formato de cada uno de los mensajes (número de bits que constituirán el tamaño de la palabra, existirá o no un bit de inicio y/o de parada, se empleará o no un bit de paridad), el tipo de lógica empleada (qué voltaje representará un cero o un uno), el orden en que serán enviados los datos y la velocidad de envío de datos.

Arduino facilita este proceso para que sólo sea necesario especificar la velocidad de envío de los datos. Esta velocidad es conocida como "baud rate" o rata de pulsos por segundo. Velocidades frecuentes de uso son 9600, 19200, 57600 y 115200.

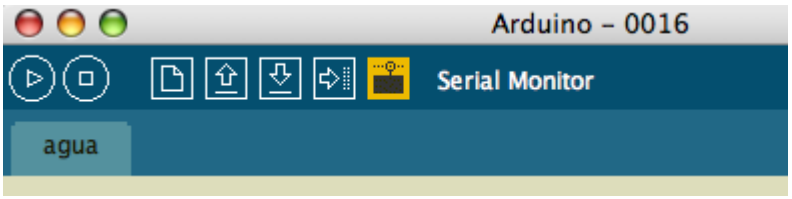


Figura 39 Monitor serial del IDE de arduino

El sistema de visión está constantemente enviando el valor del PWM que deberán tomar los servomotores, para evitar que el sistema de control se vea saturado por valores "XYXYXY…", se envían primero tres caracteres y posteriormente el valor de la coordenada en "X", se repite el mismo procedimiento con la coordenada en "Y", quedando nuestra cadena de datos de la siguiente manera "X,X,X,PWMX,Y,Y,Y,PWMY,X,X,X,PWMX….", La forma en la que se envían los datos y reciben se muestra en el anexo del código fuente de arduino.

6.1.5. Diagrama de conexión

En el siguiente diagrama se muestra como se realizan las conexiones entre los distintos dispositivos del sistema de seguimiento de objetos.

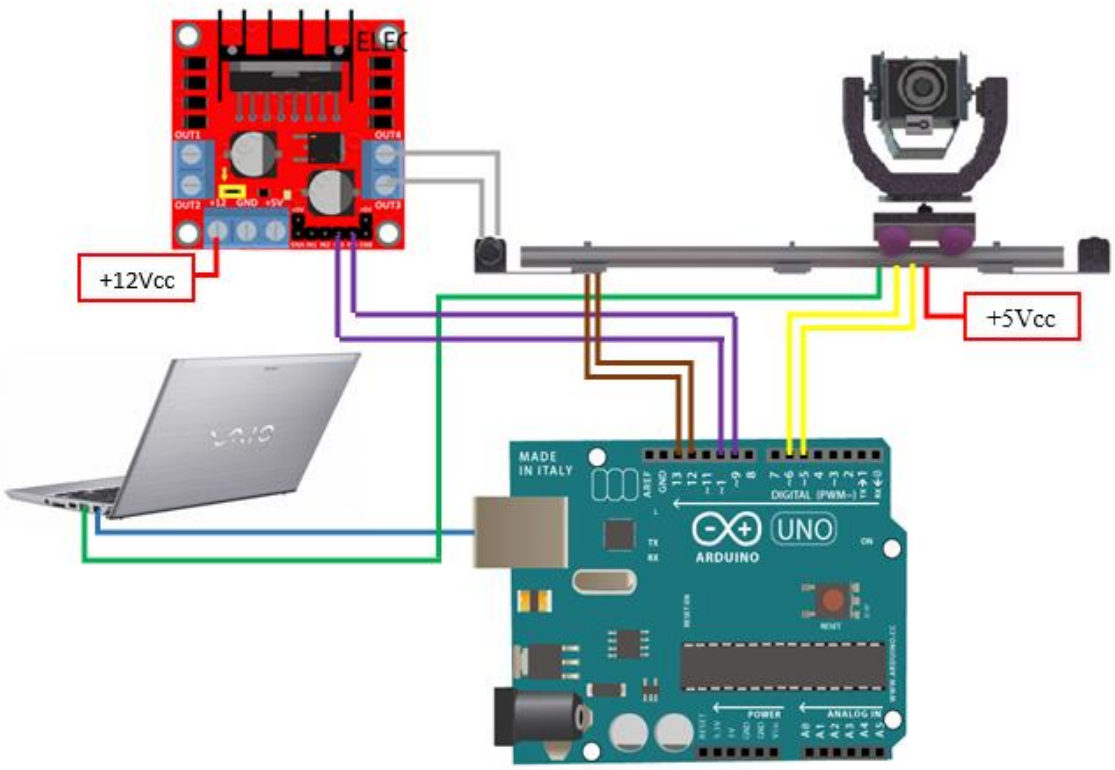


Figura 40 Diagrama de conexión del sistema completo

Conexión	Datos
Verde	Imagen enviada por el sensor de visión
Azul	Datos enviados de la PC al microcontrolador
Amarillo	Señal de PWM para ambos servomotores
Cafe	Señal de sensores de fin de carrera
Morado	Señal de giro para el motor de CD
Rojo	Alimentación
Gris	Señal de salida del puente H

Tabla 4 Condigo de colores de conexión

6.1.6. Sensor inductivo de fin de carrera SIEF

El sensor inductivo es un dispositivo de proximidad eléctrico con un display indicador de estado. El detector de posición inductivo incluye un oscilador constituido por un circuito de resonancia paralelo con bobina y un amplificador. El campo electromagnético está dirigido hacia el exterior gracias al núcleo ferrítico de la bobina.

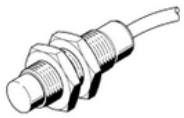


Figura 41 Sensor inductivo SIEF

Conexión eléctrica

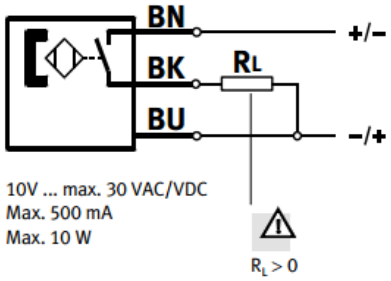


Figura 42 Conexión eléctrica

BN- Marrón
BK- Negro
BU- Azul
RL- Carga

Datos técnicos

Tensión de funcionamiento 10 ... 30 VAC/VDC

Medición de la intensidad de funcionamiento 500 mA

Distancia nominal de conmutación 10 mm

Tiempo de conmutación ≤ 0,5 ms

Tiempo de desconexión ≤ 0,03 ms

Repetibilidad ± 0,1 mm

6.1.7. Modificación de los servomotores

La selección de los servomotores se llevó a cabo con referencia al valor del torque obtenido en los cálculos del paneo y barrido de la cámara, los servomotores utilizados se muestran a continuación.

SPG5485A Rotación estándar

Figura 43 Servomotor modificado

6.1.8. Características

El soporte del servomotor es de aluminio 6061-T6

El eje es de acero inoxidable con un soporte de rodamiento de bolas con una carga máxima de 200 Lb.

El potenciómetro de precisión está protegido por una carcasa de aluminio montado sobre el eje de salida final para proporcionar la información del posicionamiento.

Para poder medir la posición del engrane exterior, se modificaron los servomotores para conectar el potenciómetro al engrane exterior y así poder medir la posición respecto a este. Internamente los engranes del servomotor tienen un tope mecánico el cual evita que el motor gire más de 180° y así evitar que el potenciómetro se pueda dañar.

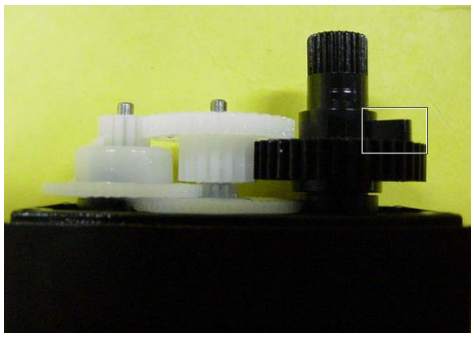


Figura 44 tope mecánico

Se removió este tope mecánico del engrane principal del servomotor quedando como se muestra en la figura 45.

Figura 45 Tope removido

Con esta modificación nuestro servomotor puede girar más de 180°, posteriormente se desoldó el potenciómetro para poder conectarlo al eje exterior de nuestro engrane.

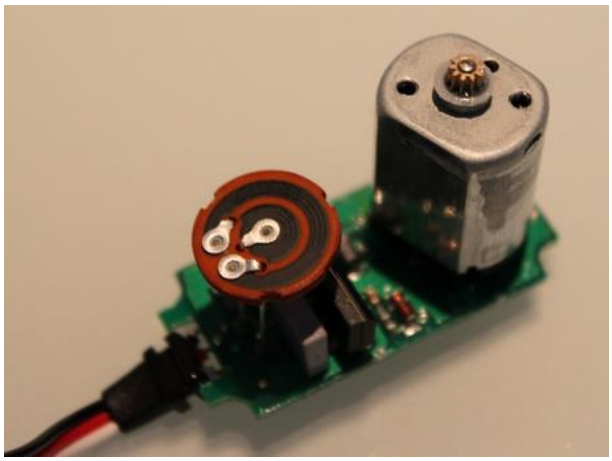


Figura 46 Potenciómetro interno

Finalmente se soldó el potenciómetro quedando de manera externa al servomotor como se muestra en la figura 47.

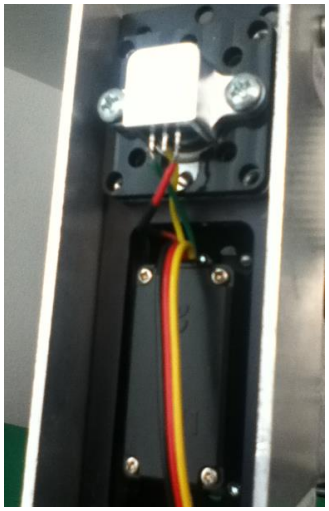


Figura 47 Potenciómetro externo

Dimensiones del soporte del servomotor

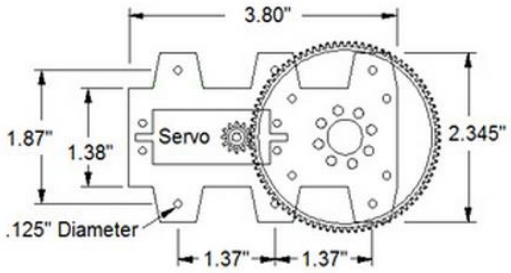


Figura 48 Dimensiones del soporte tomado de (Robotzone, 2014)

Capítulo 7
Manufactura

7. Manufactura/Implementación

La estructura y eslabones del robot se maquinaron mediante el diseño CAD creado en Solidworks, la manufactura se llevó a cabo con la ayuda de la fresa CNC Hass que la escuela tiene disponible en el taller de máquinas y herramientas.

Figura 49 CAD del robot

Los planos de los eslabones fueron exportados al software de manufactura Mastercam.

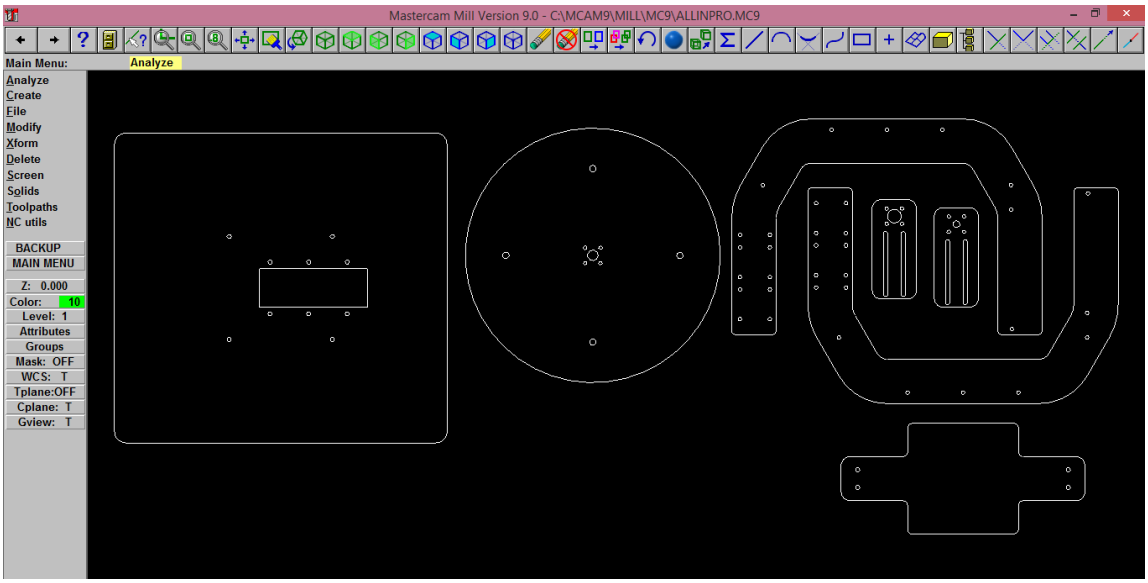


Figura 50 Espacio de trabajo de Mastercam

La herramienta utilizada para el maquinado de contornos y barrenos fue un cortador Weston de 1/8” HSS (acero de alta velocidad).

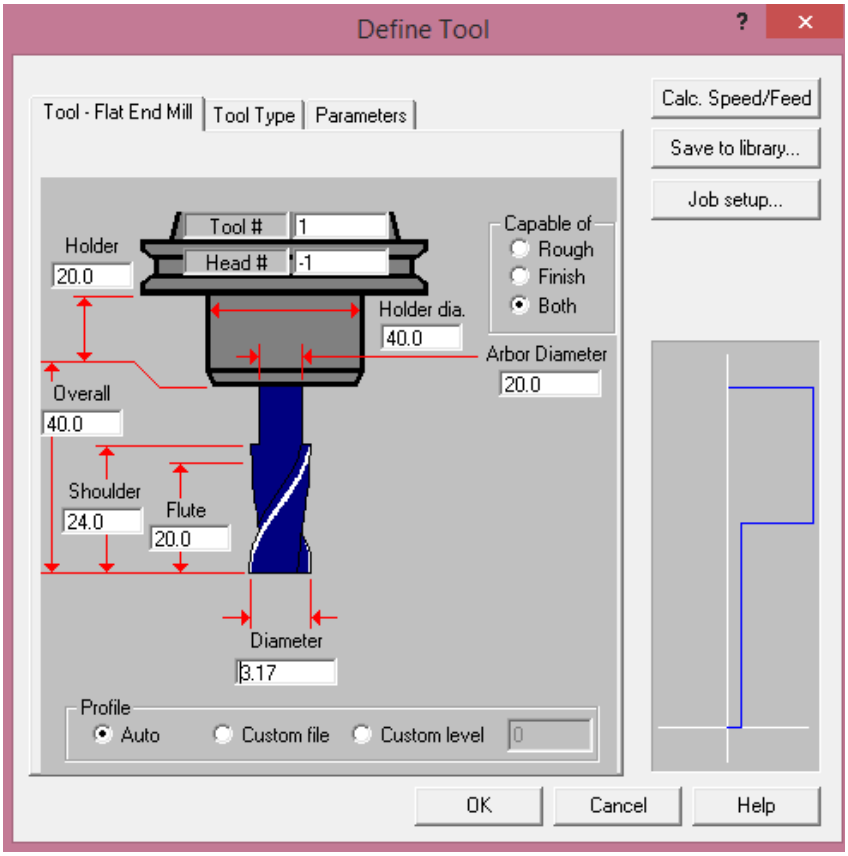


Figura 51 Parámetros de la herramienta

Los parámetros del corte de contornos se muestran en la figura 52, se utilizó la función de compensación de radio de herramienta para obtener una mejor precisión en el corte y así respetar las dimensiones del diseño CAD.

La velocidad de corte es proporcionada por el fabricante, para el aluminio 6063 se especifica un avance por diente f de 0.04 mmpr y una velocidad de corte de 75 a 400 m/min, se seleccionó una velocidad de corte de 75 m/min.

La fórmula para obtener las revoluciones por minuto del cortador es

$$N = \frac{(v_c)(1000)}{(\pi)(D)}$$

Donde:

v_c= Velocidad de corte (m/min)

D= diámetro de la herramienta

Sustituyendo la ecuación con los valores proporcionados, nos queda que las revoluciones por minuto de nuestra herramienta será de

$$N = \frac{(v_c)(1000)}{(\pi)(D)} = \frac{(75)(1000)}{(\pi)(3.175)} = 7520\ RPM$$

Para obtener el avance de la herramienta se utilizó la siguiente ecuación

$$F = (f)(z)(N)$$

Donde:

Z= Corresponde al número de dientes de la herramienta

f= Avance por diente (mmpr)

Sustituyendo la ecuación no da como resultado un avance de la herramienta de

$$F = (0.04)(4)(7520) = 1203\ mm/min$$

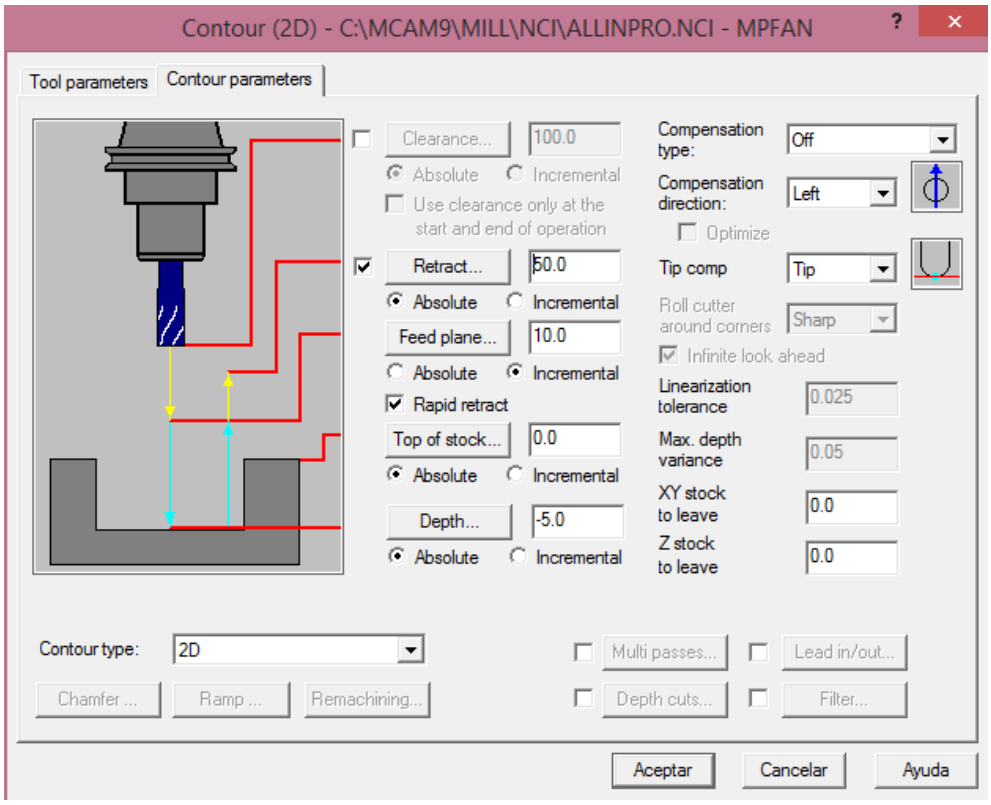


Figura 52 Parámetros de corte

Los eslabones fueron colocados de tal forma que se aprovechara toda la placa de aluminio para así desperdiciar el mínimo de material, se trazó la trayectoria de corte con las herramientas de chain y drill que provee mastercam, quedando la distribución de las piezas como se muestra en la figura 53.

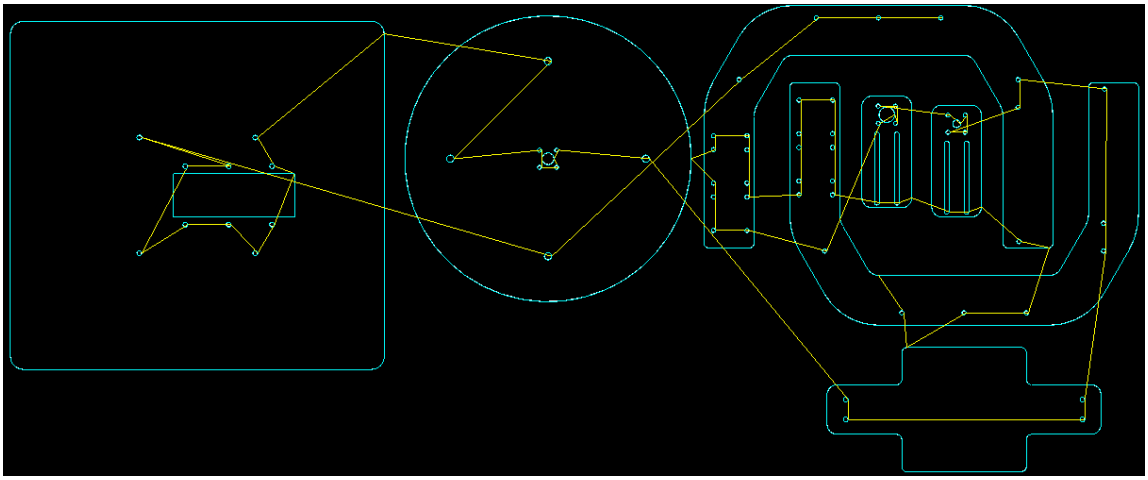


Figura 53 Trayectoria del cortador

La simulación del maquinado verifico que la herramienta y las alturas que se ingresaron a los parámetros de corte fueran correctas, se puede observar que todas las piezas se generan correctamente y los cortes no se enciman pudiendo generar problemas con el cortador al momento de que avance.

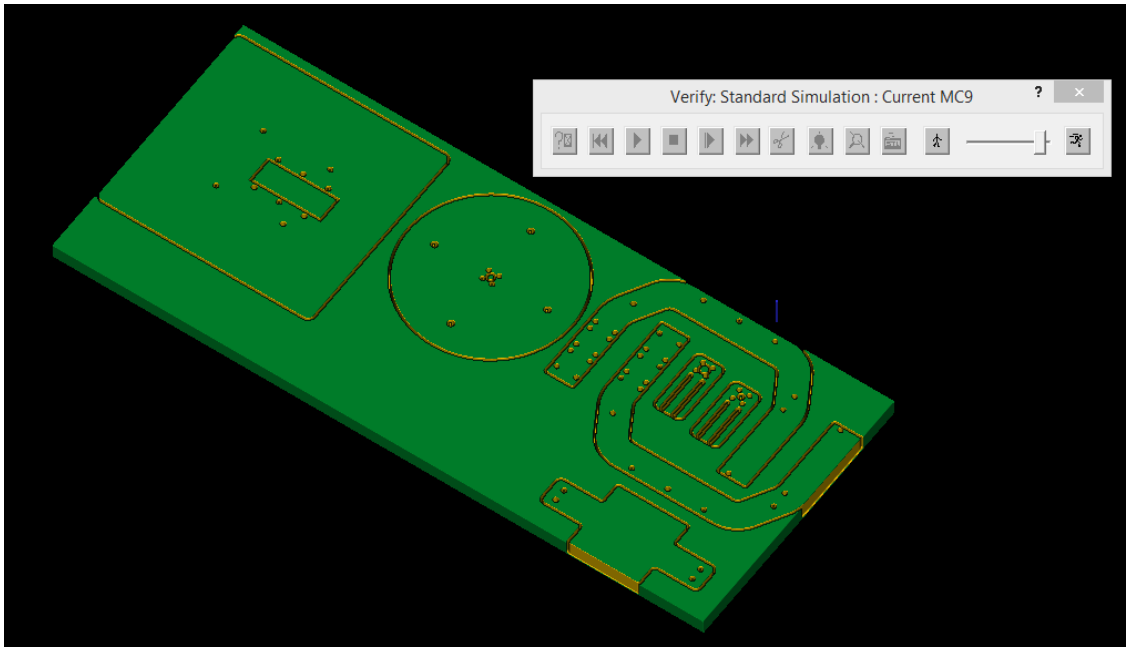


Figura 54 Verificación del Maquinado

Capítulo 8
Análisis de resultados

8. Análisis de Resultados

En este capítulo se presentan los resultados que el sistema de visión artificial nos entrega al implementar las herramientas y técnicas de procesamiento de imágenes utilizadas dentro de nuestro código en Visual Studio.

- Adquisición de la imagen.

La imagen es adquirida en formato RGB como se muestra en la figura 55.

Figura 55 Imagen original

- Identificación del color del objeto a seguir.

La imagen adquirida es convertida a formato HSV para poder identificar exactamente el color del objeto que se desea seguir como se muestra en la fig. 56.

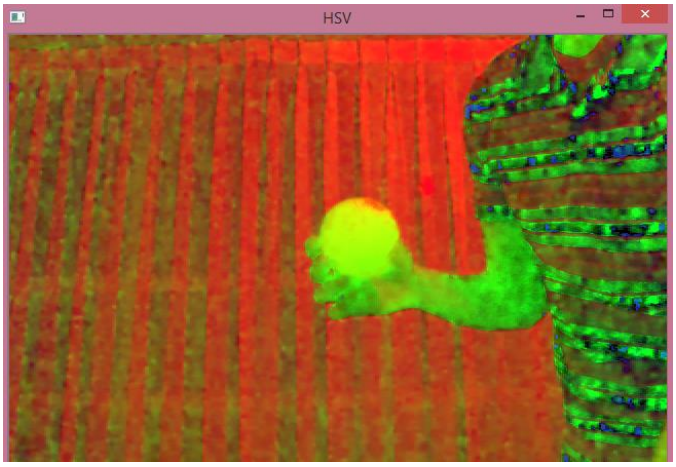


Figura 56 Imagen en modelo HSV

La imagen de la figura 57 nos muestra el resultado de la umbralización y la binarización, se encuentra toda en color blanco, ya que aún no se aplica ningún filtro para detectar el color deseado.

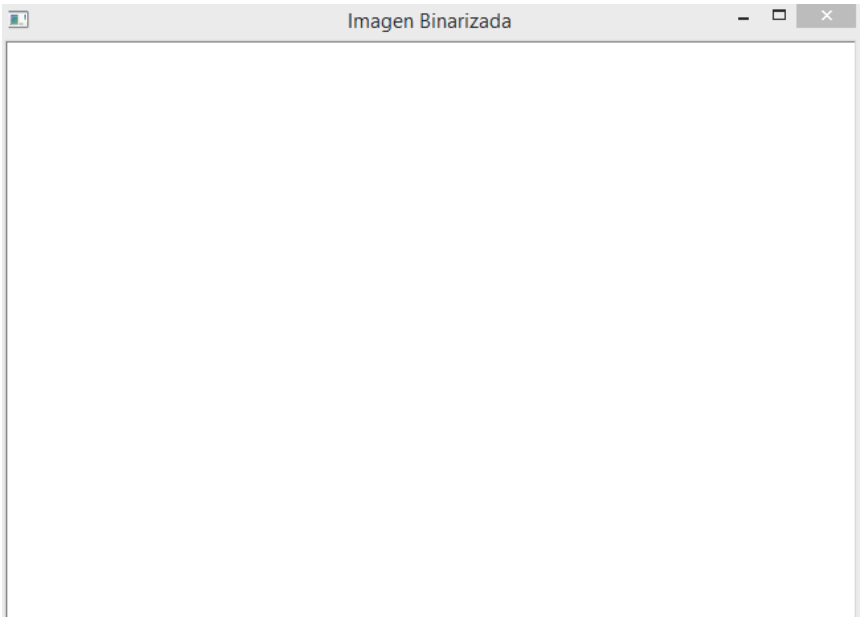


Figura 57 Imagen binarizada

Se realiza una umbralización de las tres capas de la imagen para filtrar solo el color, la saturación y el brillo deseado como se muestra en la fig.58.

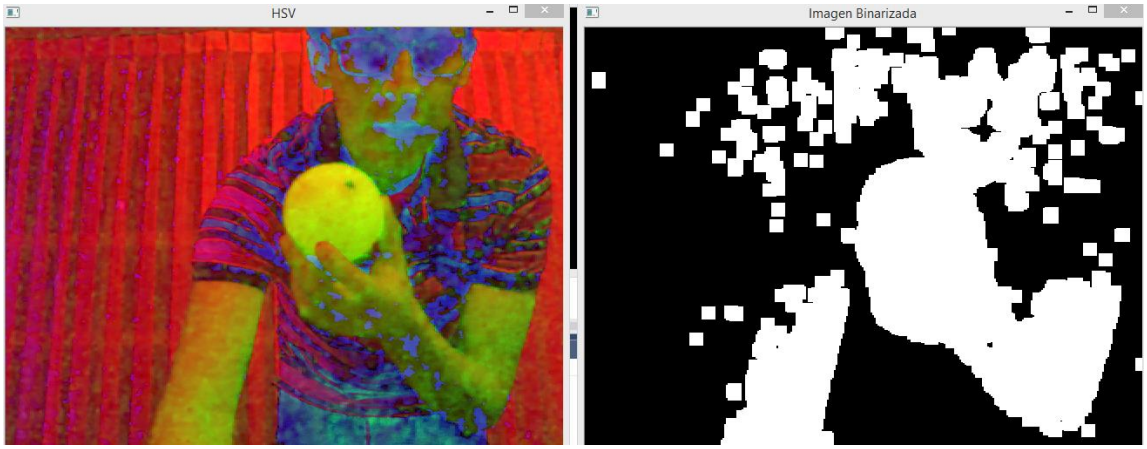


Figura 58 Umbrlización

Cuando comenzamos a variar los rangos de las tres componentes, color, saturación y brillo, el filtrado del color se comienza a observar en nuestra imagen binarizada, pudiendo así comenzar a detectar el objeto que se desea seguir.

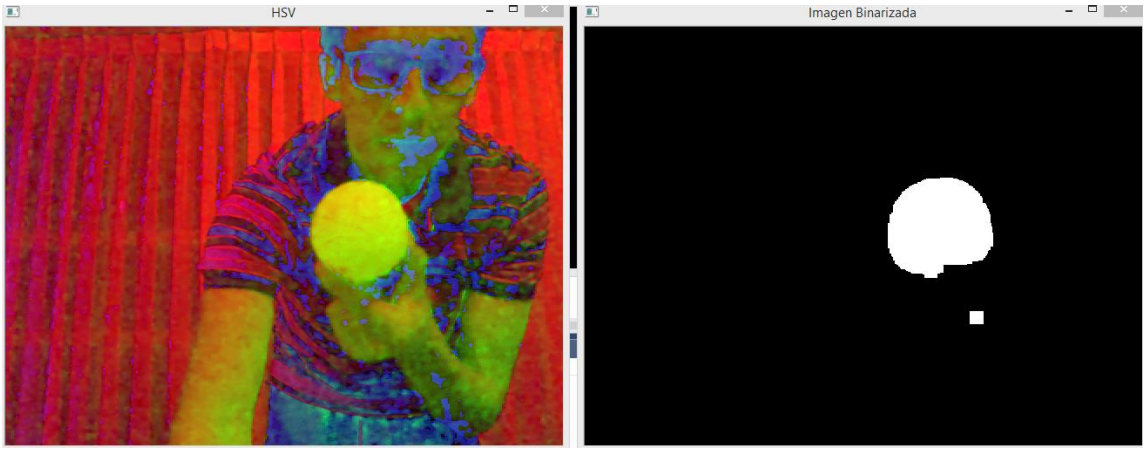


Figura 59 Color filtrado

La umbralización de la imagen se obtiene mediante rangos de los valores deseados, valores que no estén dentro de ese rango serán marcados con un cero que corresponde al color negro.

La imagen binarizada presenta algunos puntos no deseados, ya que el color deseado no se puede filtrar al 100% mediante la aplicación de rangos en HSV, estos pixeles que no pertenecen a nuestro objeto buscado son llamados ruido en nuestra imagen, para filtrar este ruido se utilizaran técnicas de operaciones morfológicas, en nuestro caso se utiliza la erosión.

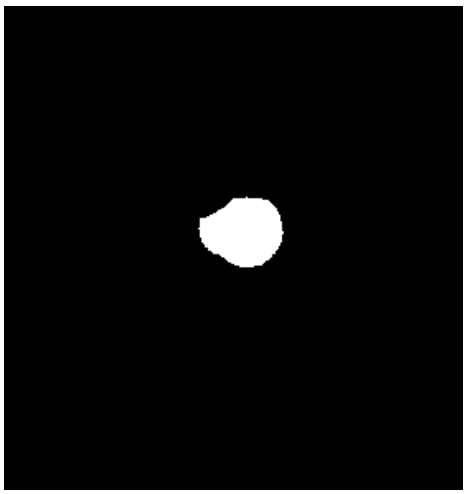


Figura 60 Imagen erosionada

Obtención del centroide del objeto a seguir y sus coordenadas, cuando el objeto es detectado se marca el centroide del objeto con las coordenadas donde se encuentra, se puede observar en la esquina superior derecha que se imprimen los valores del PWM que se envían al subsistema de control.

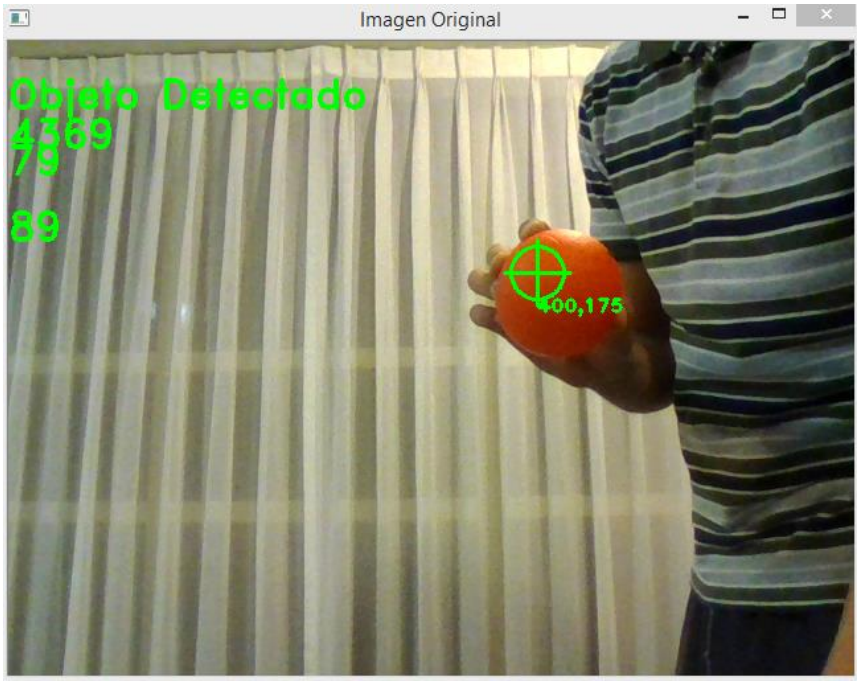


Figura 61 Coordenadas del centroide

8.1. Detección del objeto

El sistema de visión al ser implementado en lenguaje C++ se convierte en un código bastante eficiente, con iteraciones de 2.6 ms en promedio. Lo cual realiza el seguimiento del objeto de manera instantánea, sin embargo aunque la detección del objeto es en unos cuantos milisegundos, el robot no puede realizar el seguimiento a la misma velocidad ya que la respuesta de los motores llega a ser en microsegundos, esta es mucho más lenta en comparación a la respuesta del sistema de visión.

La velocidad de seguimiento que se planteó al inicio de este proyecto fue una velocidad de 0.8 m/s, el prototipo puede llegar a realizar el seguimiento del objeto a esta velocidad siempre que este se mantenga a una distancia mayor de 1.5m de separación con la cámara.

Para que un objeto pueda ser detectado el área en pixeles debe ser mayor a 400 pixeles, de esta forma se comienza el seguimiento del objeto, de lo contrario se toma como ruido y el seguimiento no se realiza.

8.2. Perturbaciones – Ruido

Algunas perturbaciones considerables que pueden llegar a afectar el sistema de seguimiento de objetos son una mala iluminación y velocidades excesivas del objeto a seguir. Una mala iluminación genera cambios repentinos de tono o brillo en el objeto que se intenta seguir, lo cual puede ocasionar que las coordenadas del objeto se pierdan. Para evitar estos problemas se utiliza un objeto que este constantemente entregando la misma cantidad de luz hacia la cámara, que no varíe su color y brillo por ejemplo una pantalla de celular.

Una velocidad excesiva sobrepasa la reacción que puede llegar a tener los servomotores llegando a hacer imposible un seguimiento del objeto deseado.

8.3. Costos

El diseño del prototipo fue pensado para una fácil manufactura y un sencillo ensamblaje tratando de aprovechar al máximo los recursos con los que se cuenta en la escuela, ya sea con herramientas y maquinaria de manufactura como es la dobladora de lámina y la fresa CNC.

Todo el maquinado del robot se realizó en la escuela por lo que no genero algún costo de manufactura, el material utilizado en el maquinado del robot se muestra a continuación

Material	Dimensiones	Costo
Placa de aluminio No.12	1.5m X 1.5m	$350
Tubo calibrado	15.9X 1.24X 3660mm	$155
Perfil de aluminio en C	1.6 X 28.6 X12.7X 3660mm	$75
Perfil de aluminio en L	1m	$40
Tornillo 6-32 de 1” de largo	50 unidades	$20
Tornillo 6-32 de ¼” de largo	50 unidades	$15
Tuerca 6-32	50 unidades	$10
Canal de aluminio	30cm	$75
	Total:	$740

El material adicional y dispositivos que se usaron en el prototipo se muestran a continuación

Material	Costo
Webcam	$150
Servomotores	$3450
Motor CD	$330
Banda dentada 5mm de paso	$400
Poleas dentadas de aluminio 5mm de paso	$120
Sensores de presencia magnéticos	$130
Conectores M12 hembra	$80
Conectores M12 macho	$80
Arduino UNO	$250
Driver lm298	$150
Botonera	$45
Fuente de alimentación	$350
Cable de 5 hilos	$70
Envíos	$300
Total:	$5905

Debido a que algunos componentes del prototipo fueron adquiridos en el extranjero, los costos de este pueden llegar a variar dependiendo el precio del dólar. El algoritmo de visión artificial al ser implementado en lenguaje C++ no genero tampoco costo alguno para la realización del prototipo.

Por lo tanto el costo estimado de nuestro prototipo es alrededor de $6645 mmx.

Trabajo futuro

Un diseño nunca está terminado, siempre hay algo que mejorar y este trabajo no es la excepción.

Este proyecto muestra las bases de un sistema de seguimiento con visión artificial, todo el proyecto puede tener mejoras tanto del software como del hardware, se puede mejorar el algoritmo de visión artificial para que no solo detecte colores, sino que también detecte formas más complejas cómo una persona caminando o incluso reconocimiento facial.

El sistema mecánico también puede mejorar mediante el uso de actuadores más robustos capaces de realizar movimientos a mayor velocidad y que acepten mayores cargas en el efector final.

Se puede buscar la implementación de otro tipo de sistema de control como un PID, y dispositivos que procesen la información de manera paralela para que la precisión del robot incremente.

La idea del seguimiento no solo se puede aplicar para la cinematografía, se pueden buscar nuevas aplicaciones por ejemplo de seguridad en el hogar, en la industria, detección de fallas, etc.

Conclusiones

El proceso de diseño es muy complejo ya que no solamente comienza en los libros y manuales sino que comienza desde la imaginación y creatividad de la persona que desarrolla el proyecto, de esta manera un proyecto puede estar pensado de diversas formas pero aun así obtener el mismo resultado.

He llegado a la conclusión de que el verdadero reto de este proyecto es poder integrar todas las partes de este, de manera que los resultados finales sean lo deseados mediante la utilización de herramientas óptimas de ingeniería que hemos adquirido a lo largo de nuestra carrera.

Al realizar el diseño mecánico es importante considerar el proceso de manufactura que seguirán las piezas, pues al comenzar la construcción mecánica nos enfrentamos a problemas ocasionados por las características y limitaciones de la maquinaria.

Por otra parte fue de gran ayuda el utilizar Software que permite realizar el maquinado de forma virtual y de esta manera se evitan errores de precisión, desperdicios y retrabajo.

Así como existen herramientas que facilitan el trabajo también es importante no descartar servicios proporcionados por talleres de manufactura y fabricantes de circuitos impresos que pueden agilizar, economizar y mejorar el trabajo, debido a que son proporcionados por talleres especializados.

Al desarrollar la conexión entre los subsistemas, la pruebas de transmisión de datos mediante comunicación serial se vieron afectadas debido a que se enviaban los datos de manera secuencial sin un orden, se tuvo que crear un método para enviar los datos y así evitar que la información se perdiera y que los motores recibieran una señal de PWM errónea.

El uso de servomotores permitió la simplificación del diseño del controlador y de la electrónica del sistema, desafortunadamente pienso que esta aparente simplificación tiene una enorme repercusión en el desempeño del sistema mecánico, ya que la resolución de los movimientos del robot se vio afectada.

El Prototipo de sistema para seguimiento de objetos con visión artificial realiza el seguimiento de un objeto de un color previamente cargado en el sistema, siempre que exista una adecuada iluminación, y el fondo sea de color diferente al del objeto que se planea seguir.

Es evidente que falta mucho por investigar sobre el tema, algoritmos de visión, control, mecánica, etc., pero espero que el presente trabajo sirva de base para que otros alumnos retomen el tema del seguimiento de objetos y lo expandan. Por último quisiera resaltar el hecho de que este prototipo de seguimiento de objetos es la base para que otras personas desarrollen nuevas aplicaciones derivadas de este trabajo y puedan utilizar las herramientas para proponer soluciones tecnológicas a problemas ya sea en la industria, seguridad, entretenimiento, etc.

Referencias

A. Yilmaz, O. J. (2006). *Object tracking.* ACM Comput.

Arduino. (2014). Obtenido de http://arduino.cc/en/Main/ArduinoBoardUno

Atmel. (2009). *http://www.atmel.com.* Obtenido de http://www.atmel.com/Images/doc8161.pdf

Azuela, J. H. (2013). *VISION ARTIFICIAL.* RA-MA.

Basu, A. (1993). Computer Vision: Systems, Theory and Applications. En A. Basu. World scientific.

Daniel, C. H. (2009). *Sistema de visión artificial para la navegación d eun robot móvil.*

Diaz, M. (2011). Aleaciones de aluminio.

Festo. (2013). Sistemas de vision.

Hibbeler, R. (2005). *Mecanica vectorial para ingenieros.* Pearson.

Jair, H. C. (2009). *Dispositivo automatizado para el control d eposicion de una marcadora mediante vision artificial.*

Lu, D. Z. (2001). *Segmentation of moving objects in image sequence: A review. Circuits, Systems and Signal Process.*

Marcos, A. G. (2006). Tecnicas y algoritmos de vision artificial. En A. G. Marcos.

OpenCV. (2014). *http://docs.opencv.org*. Obtenido de http://docs.opencv.org/master/modules/imgproc/doc/structural_analysis_and_shape_descriptors

OpenCV. (2014). *http://opencv.org*. Obtenido de http://opencv.org/documentation.html

Raúl, T. P. (2010). *Sistema de visión para la supervisión del control de calidad en textiles.*

STMicroelectronics. (2000). *http://www.datasheetcatalog.com.* Obtenido de http://www.datasheetcatalog.com/datasheets_pdf/L/2/9/8/L298N.shtml

Suzuki. (1985). *http://wenku.baidu.com.* Obtenido de http://wenku.baidu.com/view/6cb52ede360cba1aa811dad5.html

Technology, A. (2012). Iron Dome Air Defence missile System.

Wheeler, P. (2006). *Cinematografia.* Omega.

Apéndices Anexos

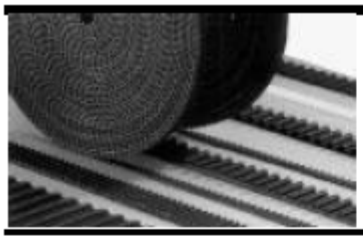
1. Manual “Rodamientos FAG” pag.76.

Rodamientos FAG rígidos de bolas de una hilera

Los rodamientos pueden alcanzar una duración de vida ilimitada, si $C_0/P_0 \geq 8$, ver Página 61.

Eje	Dimensiones							Peso	Capacidad de carga		Velocidad límite	Velocidad de referencia	Denominación abreviada	Medidas auxiliares			
	d	D	B	r_s mín.	H ≈	H_1 ≈	J ≈		din. C	stat. C_0			Rodamiento	D_1 mín.	D_2 máx.	r_a máx.
mm								kg	kN		min^{-1}		FAG	mm		
3	10	4	0,15	7,7	8,2	5	0,001	0,84	0,22	53000	67000	623	4,4	8,6	0,15
3	10	4	0,15	7,7	8,2	5	0,001	0,84	0,22	45000	67000	623-2Z	4,4	8,6	0,15
3	10	4	0,15	7,7	8,2	5	0,001	0,84	0,22	32000	67000	623-2RS	4,4	8,6	0,15
4	13	5	0,2	10,5	11,2	7	0,003	1,29	0,49	45000	53000	624	5,8	11,2	0,2
4	13	5	0,2	10,5	11,2	7	0,004	1,29	0,49	38000	53000	624-2Z	5,8	11,2	0,2
4	13	5	0,2	10,5	11,2	7	0,003	1,29	0,49	26000	53000	624-2RS	5,8	11,2	0,2
4	16	5	0,3	12,5	13,4	8,5	0,006	1,73	0,67	43000	43000	634	6,4	13,6	0,3
4	16	5	0,3	12,5	13,4	8,5	0,006	1,73	0,67	36000	43000	634-2Z	6,4	13,6	0,3
4	16	5	0,3	12,5	13,4	8,5	0,006	1,73	0,67	24000	43000	634-2RS	6,4	13,6	0,3
5	16	5	0,3	12,5	13,4	8,5	0,005	1,32	0,44	43000	43000	625	7,4	13,6	0,3
5	16	5	0,3	12,5	13,4	8,5	0,005	1,32	0,44	36000	43000	625-2Z	7,4	13,6	0,3
5	16	5	0,3	12,5	13,4	8,5	0,005	1,32	0,44	24000		625-2RS	7,4	13,6	0,3
5	19	6	0,3	15,5	16,7	10,8	0,008	2,55	1,04	38000	40000	635	7,4	16,6	0,3
5	19	6	0,3	15,5	16,7	10,8	0,009	2,55	1,04	32000	40000	635-2Z	7,4	16,6	0,3
5	19	6	0,3	15,5	16,7	10,8	0,008	2,55	1,04	22000		635-2RS	7,4	16,6	0,3
6	19	6	0,3	15,5	16,7	10,6	0,008	2,55	1,04	38000	38000	626	8,4	16,6	0,3
6	19	6	0,3	15,5	16,7	10,6	0,009	2,55	1,04	32000	38000	626-2Z	8,4	16,6	0,3
6	19	6	0,3	15,5	16,7	10,6	0,008	2,55	1,04	22000		626-2RS	8,4	16,6	0,3
7	19	6	0,3	15,5	16,7	10,6	0,007	2,55	1,04	38000	38000	607	9	17	0,3
7	19	6	0,3	15,5	16,7	10,6	0,009	2,55	1,04	32000	38000	607-2Z	9	17	0,3
7	19	6	0,3	15,5	16,7	10,6	0,007	2,55	1,04	22000		607-2RS	9	17	0,3

2. Manual “Master de bandas industriales” pag. 52.



Bandas Sincrónicas Con Fin (Extremo Abierto)

Gates tiene disponibles los siguientes tipos de bandas sincrónicas de Extremo Abierto:

I. Poly Chain GT

II. PowerGrip GT / HTD

III. PowerGrip Timing

IV. Synchro-Power

Estas bandas están diseñadas para transmisiones industriales que requieran de piñón y cremallera únicamente.

Disponibilidad:

- Solamente se tienen disponibles los pasos y los anchos de bandas mostrados en las tablas correspondientes.

Poliuretano con refuerzo de Kevlar

Poly Chain GT Paso 8 mm

DESCRIPCION	ANCHO (mm)	Longitud Mínima (m)
LL8M012GT	12	3.05

Neopreno con refuerzo de Fibra de Vidrio

PowerGrip GT Paso 2 mm

DESCRIPCION	ANCHO (mm)	Longitud Mínima (m)
LL2MR04	4	91.5
LL2MR06	6	91.5
LL2MR09	9	91.5

PowerGrip GT Paso 3 mm

DESCRIPCION	ANCHO (mm)	Longitud Mínima (m)
LL3MR06	6	91.5
LL3MR09	9	76.2
LL3MR15	15	45.7

PowerGrip GT Paso 5 mm

DESCRIPCION	ANCHO (mm)	Longitud Mínima (m)
LL5MR09	9	91.4
LL5MR15	15	68.6
LL5MR25	25	38.1

PowerGrip GT Paso 8 mm

DESCRIPCION	ANCHO (mm)	Longitud Mínima (m)
LL8MR20	20	55.5
LL8MR30	30	91.5
LL8MR50	50	91.5
LL8MR85	85	91.5

3. Código fuente del subsistema de control implementado en arduino

```
#include <Servo.h>

Servo servox;
Servo servoy;
int servoPosition = 100;
int incomingByte = 0;   // Dato recibido
//*************************************************************
const int ledPin1 =  9;      //Pin control puente H
const int ledPin2 =  10;           //Pin control puente H
const int ledPin3 =  2;    //Sensor1
const int ledPin4 =  3;    //Sensor2
const int motor=11;

int sensor1=0;
int sensor2=0;
int histeresis=2;
int bandera1=1;
int bandera2=1;
void setup()
{
  Serial.begin(9600); //Habilita el puerto serial a 9600 bps
  servox.attach(5);
  servoy.attach(6);
  servox.write(servoPosition);
  servoy.write(servoPosition);
  //delay(6000);
  pinMode(ledPin1, OUTPUT); //modo salida ledpin1
  pinMode(ledPin2, OUTPUT); //modo salida ledpin2

  pinMode(ledPin3, INPUT); //modo salida ledpin3
  pinMode(ledPin4, INPUT); //modo salida ledpin4
  pinMode(motor,OUTPUT);

  digitalWrite(ledPin1, LOW);
  digitalWrite(ledPin2, LOW);
  analogWrite(motor,110);
}

int contx=0,bandx=0;
int conty=0,bandy=0;

void loop()
{
  if (Serial.available() > 0) {

    incomingByte = Serial.read();
    if (bandx==0 && incomingByte == 'x')
    {
      //if(incomingByte == 'x')
        contx++;
      //else
      //  contx=0;

      if(contx>=3)
      {
        bandx=1;
      }
```

```
    }
  }else if (bandy == 0 && incomingByte == 'y')
  {
    conty++;
    if (conty>=3)
    {
      bandy = 1;
    }
  }
  else
  {
    if(bandx==1)
    {
      servox.write(incomingByte);
      servoPosition=incomingByte;
      bandx=0;
      contx=0;
    }
    if(bandy == 1)
    {
      servoy.write(incomingByte);
      //servoPosition=incomingByte;
      bandy=0;
      conty=0;
    }
  }

}
  sensor2= digitalRead(ledPin3);
  sensor1= digitalRead(ledPin4);


if(servoPosition<=90 && sensor1==LOW)
      {
              histeresis=1;
              bandera1=1;
              bandera2=0;
              digitalWrite(ledPin1, HIGH);//Giro a la X
              digitalWrite(ledPin2, LOW);

      }
else if(servoPosition>95&&servoPosition<=110)
      {
              histeresis=2;
              digitalWrite(ledPin1, LOW);
              digitalWrite(ledPin2, LOW);
      }
else if(servoPosition>115 && sensor2==LOW)
      {
              bandera1=0;
              bandera2=1;
              histeresis=3;
              digitalWrite(ledPin1, LOW);
              digitalWrite(ledPin2, HIGH);
      }
```

```
  else if(servoPosition>90&&servoPosition<=95&&histeresis==2)
        {
              digitalWrite(ledPin1, LOW);//Giro a la X
              digitalWrite(ledPin2, LOW);
        }
  else if(servoPosition>110&&servoPosition<=115&&histeresis==3&&sensor2==LOW)
        {
              digitalWrite(ledPin1, LOW);//Giro a la X
              digitalWrite(ledPin2, HIGH);
        }
  else if(servoPosition>110&&servoPosition<=115&&histeresis==2)
        {
              digitalWrite(ledPin1, LOW);//Giro a la X
              digitalWrite(ledPin2, LOW);
        }
  if(sensor1==HIGH && bandera1==1)
        {
             digitalWrite(ledPin1, LOW);//Giro a la X
             digitalWrite(ledPin2, LOW);
        }
  if(sensor2==HIGH && bandera2==1)
        {
             digitalWrite(ledPin1, LOW);//Giro a la X
             digitalWrite(ledPin2, LOW);
        }

}
```

4. Código fuente del sistema de visión artificial

```
#include <sstream>
#include <string>
#include <iostream>
#include <opencv\highgui.h>
#include <opencv\cv.h>
#include <Windows.h>
#include <time.h>
using namespace cv;

/*int H_MIN = 9;//pelota naranja
int H_MAX = 33;
int S_MIN = 135;
int S_MAX = 256;
int V_MIN = 251;
int V_MAX = 256;*/
//captura del ancho y alto de la imagen
const int ALTO_IM = 480;
const int ANCHO_IM = 640;
//numero maximo de objetos detectados
const int MAX_NUM_OBJECTS = 50;
//area minima y maxima del objeto
const int MIN_OBJE_AREA = 20 * 20;
const int MAX_OBJE_AREA = ALTO_IM*ANCHO_IM / 1.5;
//nombres de las ventanas a usar
const string ventana = "Imagen Original";
const string ventana1 = "HSV";
const string ventana2 = "Imagen Binarizada";
//const string ventana3 = "Operaciones morfologicas";
const string trackbarVentana1 = "Trackbars";

int main(int argc, char* argv[])
{
    HANDLE hSerial = CreateFile("COM3", GENERIC_READ | GENERIC_WRITE, 0, 0, OPEN_EXISTING,
    if (hSerial != INVALID_HANDLE_VALUE)
    {


        DCB dcbSerialParams;
        GetCommState(hSerial, &dcbSerialParams);

        dcbSerialParams.BaudRate = CBR_9600;
        dcbSerialParams.ByteSize = 8;
        dcbSerialParams.Parity = NOPARITY;
        dcbSerialParams.StopBits = ONESTOPBIT;

        SetCommState(hSerial, &dcbSerialParams);
    }
    else
```

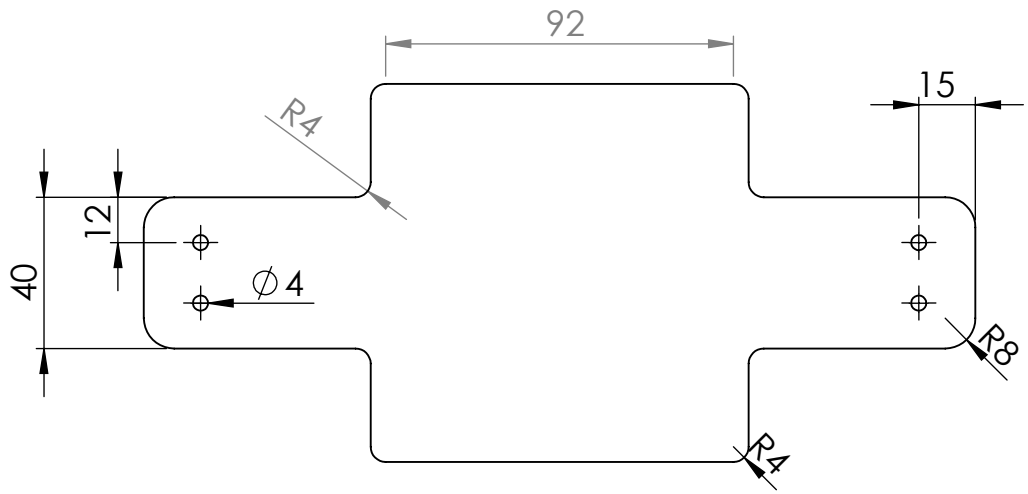
```
    {
        if (GetLastError() == ERROR_FILE_NOT_FOUND)
        {
            printf("El Puerto serial no existe! \n");
        }
        printf("Error al abrir el Puerto serial! \n");
    }

    DWORD btsIO;


    bool seguir_objeto = true;
    bool morfologia = true;
    //Matriz para guardar cada cuadro de la camara
    Mat cuadros;
    //matriz para guardar la matriz HSV
    Mat HSV;
    //matriz para guardar la imagen binarizada binarizada image
    Mat binarizada;
    //X y Y coordenadas del objeto

int x = 320, y = 240;
double refArea = 2000;
char arregloX[4] = { 'x', 'x', 'x', '0' };
char arregloY[4] = { 'y', 'y', 'y', '0' };
char arregloArea[4] = { 'A', 'A', 'A', '0' };
double errX, pwmX;
double errY, pwmY;
//double pwmA;
//funcion para crear las trackbars para el filtrado en HSV
creaTrackbars();
//captura de video para adquirir las imagenes
VideoCapture capture;
//abrir la captura de video de la webcam
//capture.open(0);//Camara laptop
capture.open(0);//Camara robot
//establecer alto y ancho de la captura
capture.set(CV_CAP_PROP_FRAME_WIDTH, ANCHO_IM);
capture.set(CV_CAP_PROP_FRAME_HEIGHT, ALTO_IM);

// Home de los motores
pwmX = 100; pwmY = 110;
variablesSerial((int)pwmX, arregloX);
WriteFile(hSerial, arregloX, 4, &btsIO, NULL);
variablesSerial((int)pwmY, arregloY);
WriteFile(hSerial, arregloY, 4, &btsIO, NULL);

waitKey(5000);
while (1){
    start = clock();
    //guarda la imagen en la matriz
    capture.read(cuadros);
    //conversion de BGR a HSV utilizando colorspace
    cvtColor(cuadros, HSV, COLOR_BGR2HSV);
    //filtrmos la imagen HSV entre los valores seleccionados
    //y la guardamos en una nueva matriz binarizada
    inRange(HSV, Scalar(H_MIN, S_MIN, V_MIN), Scalar(H_MAX, S_MAX, V_MAX), binarizada);
    //se realizan las operaciones morfologicas para eliminar ruido
```

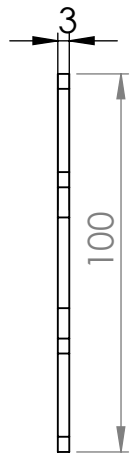
```
//se reaizan las operaciones morfologicas para eliminar ruido
if (morfologia)
    f_morfologia(binarizada);
//se manda llamar a la funcion object tracking
//esta funcion regresara las coordenadas del objeto
if (seguir_objeto)
    f_seguir_objeto(refArea, x, y, binarizada, cuadros);

errX = 320 - x;
errY = 240 - y;


//errX = (x*90/640)*1+60;
//errY = (y*90/480)*1+60;

if (errY > 0)
{
    if (pwmY >= 90)
        pwmY = pwmY - 1 * abs(errY) / 240 * 2;
}
else if (errY <= 0)
{
    if (pwmY <= 140)
        pwmY = pwmY + 1 * abs(errY) / 240 * 2;
}


//*********************************************
if (errX < 0)
{
    if (pwmX >= 80)
        pwmX = pwmX - 1 * abs(errX) / 320 * 1.5;
}

else if (errX >= 0)
{
    if (pwmX <= 130)
        pwmX = pwmX + 1 * abs(errX) / 320 * 1.5;
}

//pwmA = (320 - x + 105) * 2;

//enviamos los valores X,Y,Area por serial
variablesSerial((int)pwmX, arregloX);
//funcion serial
WriteFile(hSerial, arregloX, 4, &btsIO, NULL);

variablesSerial((int)pwmY, arregloY);
//funcion serial
WriteFile(hSerial, arregloY, 4, &btsIO, NULL);

putText(cuadros, intToString(pwmX), Point(0, 100), 2, 1, Scalar(0, 255, 0), 2);
putText(cuadros, intToString(pwmY), Point(0, 150), 2, 1, Scalar(0, 255, 0), 2);
```

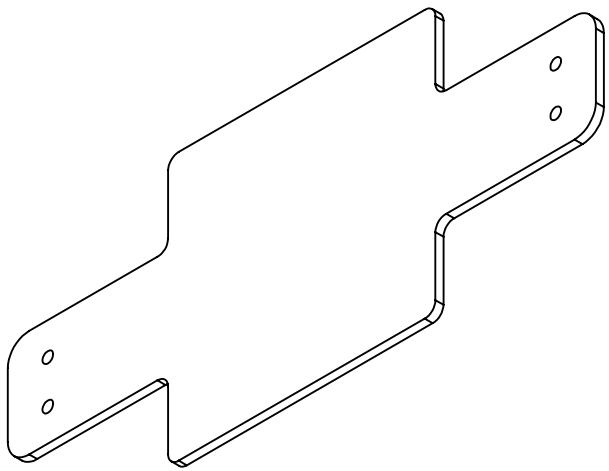
```
 //imprimimos las ventanas correspondientes
 imshow(ventana2, binarizada);
 imshow(ventana, cuadros);
 imshow(ventana1, HSV);

 char c = (char)waitKey(10);
 if (c == 27)
     break;
 switch (c)
 {
     //PAUSA
 case 'p':
     seguir_objeto = false;
     break;
     //Seeguimiento
 case 's':
     seguir_objeto = true;
     break;

        default:
            ;
        }



        //esperamos 30ms para que la pantalla se pueda actualizar
        //la imagen no aparecera si no esperamos un instante de tiempo
        waitKey(30);
    }

    finish = clock();
    duration = (double)(finish - start) / CLOCKS_PER_SEC;
    printf("%2.1f seconds\n", duration);




    return 0;
}

void f_morfologia(Mat &thresh){
    //se crean las estructuras para realizar la dilatacion y erosion
    //la mascara de erosion es de 3x3

    Mat M_erosion = getStructuringElement(MORPH_RECT, Size(3, 3));
    //la dilatacion se hace con un elemento mayor para asegurarnos que el objeto sea visible
    Mat M_dilatacion = getStructuringElement(MORPH_RECT, Size(8, 8));

    erode(thresh, thresh, M_erosion);
    erode(thresh, thresh, M_erosion);
    dilate(thresh, thresh, M_dilatacion);
    dilate(thresh, thresh, M_dilatacion);



}
```

```
void f_seguir_objeto(double &refArea, int &x, int &y, Mat binarizada, Mat &cuadros){

    Mat temp;
    binarizada.copyTo(temp);
    //vectores de salida necesarios para la funcion findContours
    vector< vector<Point> > contorno;
    vector<Vec4i> jerarquia;

    //encontramos el controno del objeto usando la funcion findContours
    //Parametros
    //1.-imagen a identificar 2.-contornos, cada contorno se almacena en un vector
    //3.-jerarquia del contorno 4.- modo 5.-metodo
    findContours(temp, contorno, jerarquia, CV_RETR_CCOMP, CV_CHAIN_APPROX_SIMPLE);
    //usamos el metodo de momentos para encontrar nuestro objeto filtrado
    refArea = 0;
```

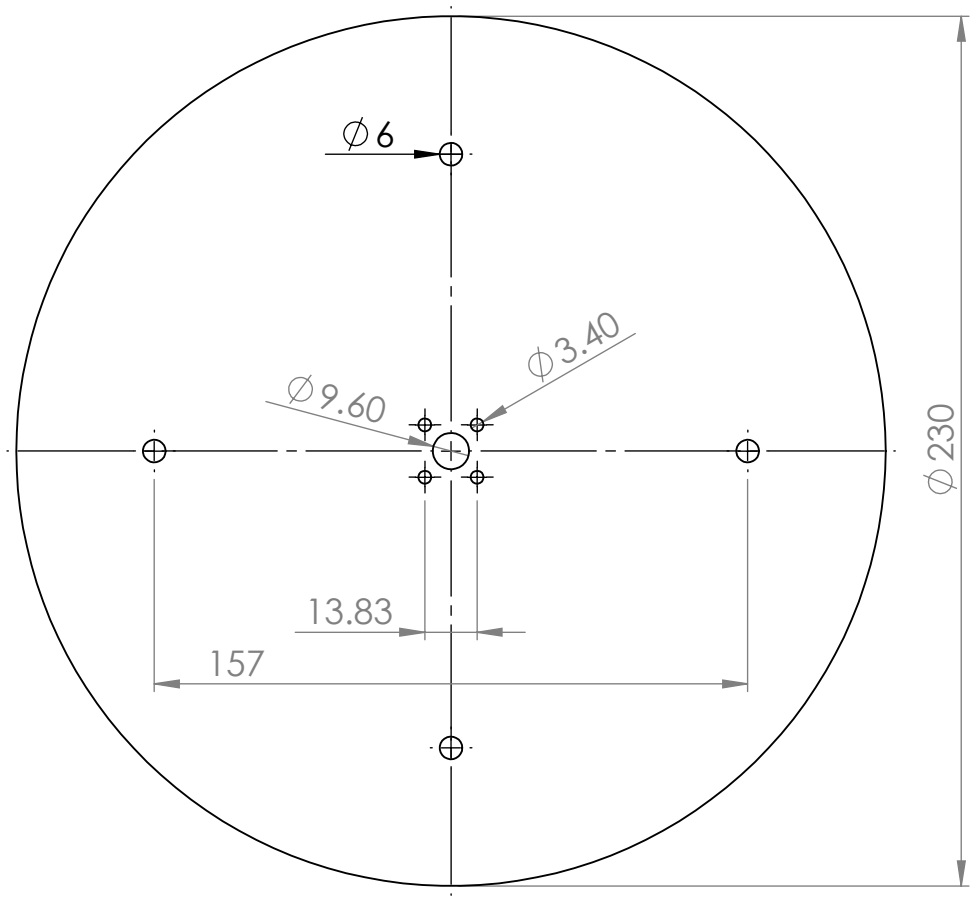
```
if (jerarquia.size() > 0) {
    int numObjects = jerarquia.size();
    //si el numero de objetos es mayor que el Max_num_objects tenemos un filtro muy ruidoso
    if (numObjects<MAX_NUM_OBJECTS){
        for (int index = 0; index >= 0; index = jerarquia[index][0]) {

            Moments moment = moments((cv::Mat)contorno[index]);
            double area = moment.m00;
            //si el area es menos a 20 x20 pixels entonces es problable que sea solo ruido
            //si el area es del tamaño de 3/2 de la imagen, es probable que el filtro este mal
            //solo queremos el objeto con el area mas grande, asi que guardamos un area de referencia
            //en cada iteracion y l comparamos con la siguiente.

            if (area>MIN_OBJE_AREA && area<MAX_OBJE_AREA && area>refArea){
                x = moment.m10 / area;
                y = moment.m01 / area;
                objectFound = true;
                refArea = area;
            }
           //imprime que el objeto se a detectado
           if (objectFound == true){
               putText(cuadros, "Objeto Detectado", Point(0, 50), 2, 1, Scalar(0, 255, 0), 2);
               putText(cuadros, intToString(refArea), Point(0, 80), 2, 1, Scalar(0, 255, 0), 2);
               //imprime la localizacion y cordenadas del objeto
               Marcador(x, y, cuadros);
           }

       }
       else putText(cuadros, "MUCHO RUIDO AJUSTA EL FILTRO", Point(0, 50), 1, 2, Scalar(0, 0, 255), 2);
   }
}
```

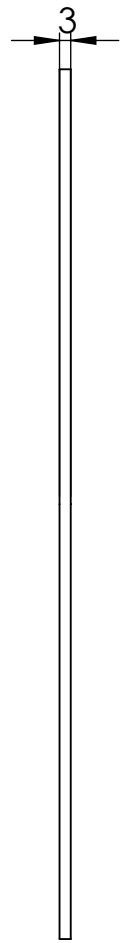




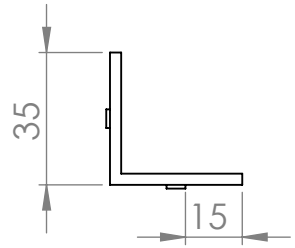


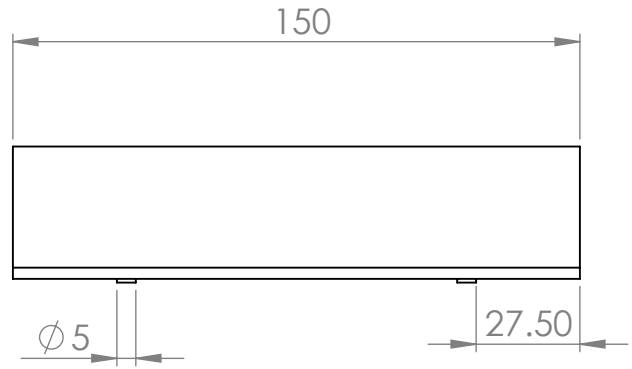
INSTITUTO POLITECNICO NACIONAL		
UNIDAD PROFESIONAL INTERDICIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS		
ALUMNO	GARCIA BARAJAS LUIS ALBERTO	
ESCALA 1:2	DIM. mm.	MATERIAL: ALUMINIO
PROTOTIPO DE SISTEMA PARA SEGUIMIENTO DE OBJETOS CON VISION ARTIFICIAL		
1	BASE CAMARA	

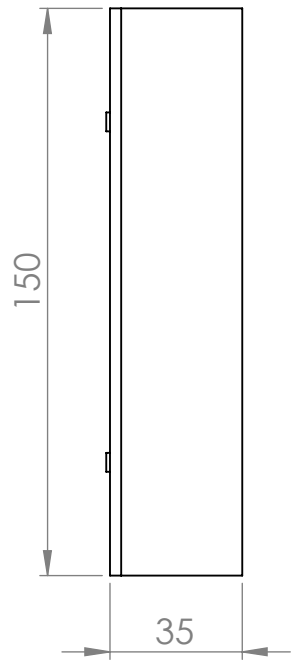


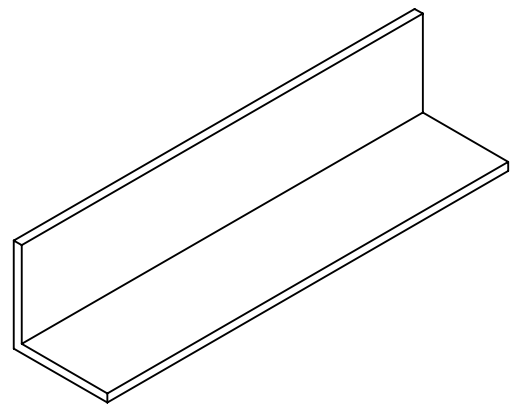


INSTITUTO POLITECNICO NACIONAL		
UNIDAD PROFESIONAL INTERDICIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS		
ALUMNO	GARCIA BARAJAS LUIS ALBERTO	
ESCALA 1:2	DIM. mm.	MATERIAL: ALUMINIO
PROTOTIPO DE SISTEMA PARA SEGUIMIENTO DE OBJETOS CON VISION ARTIFICIAL		
2	BASE CIRCULAR	

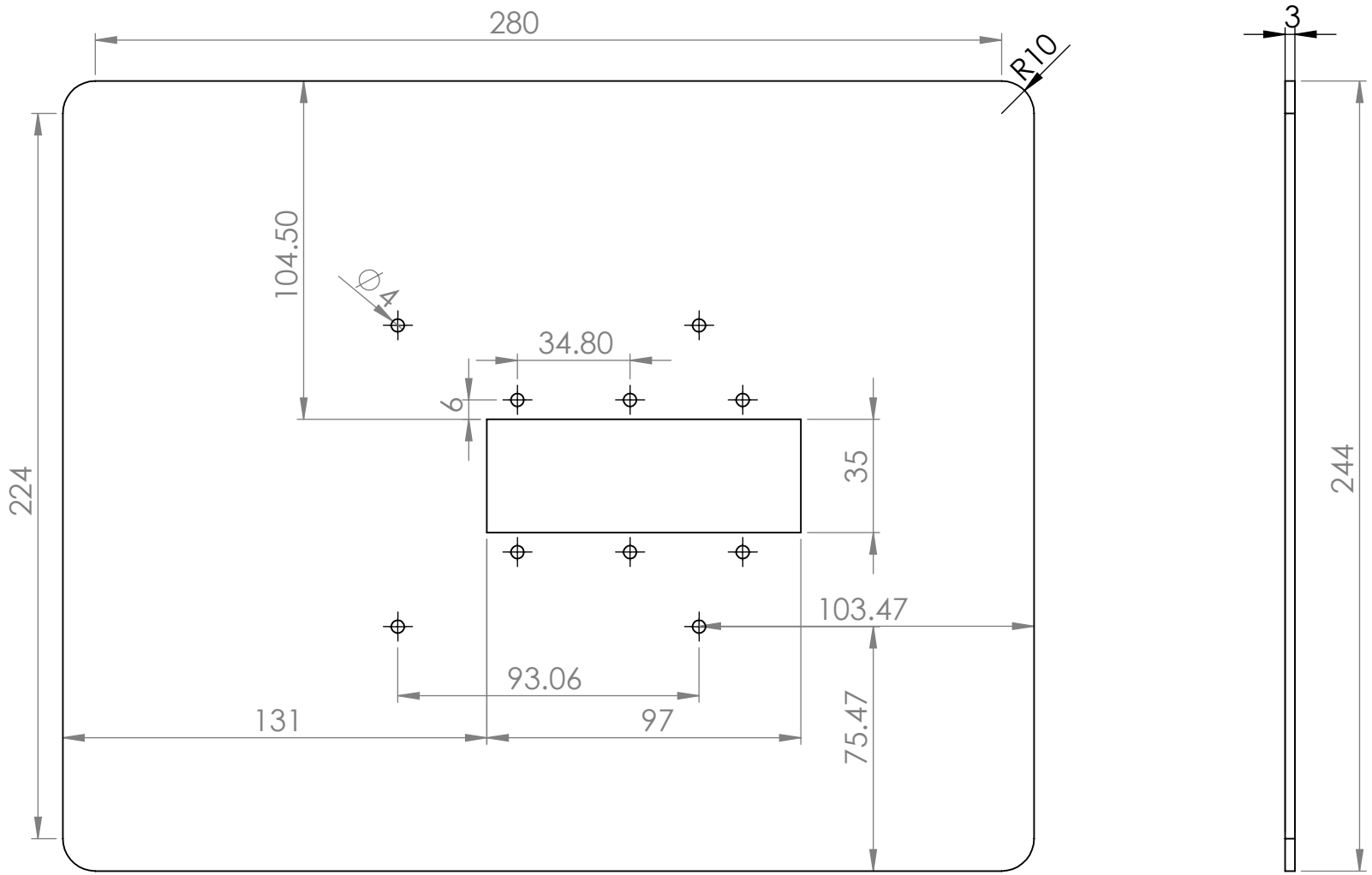




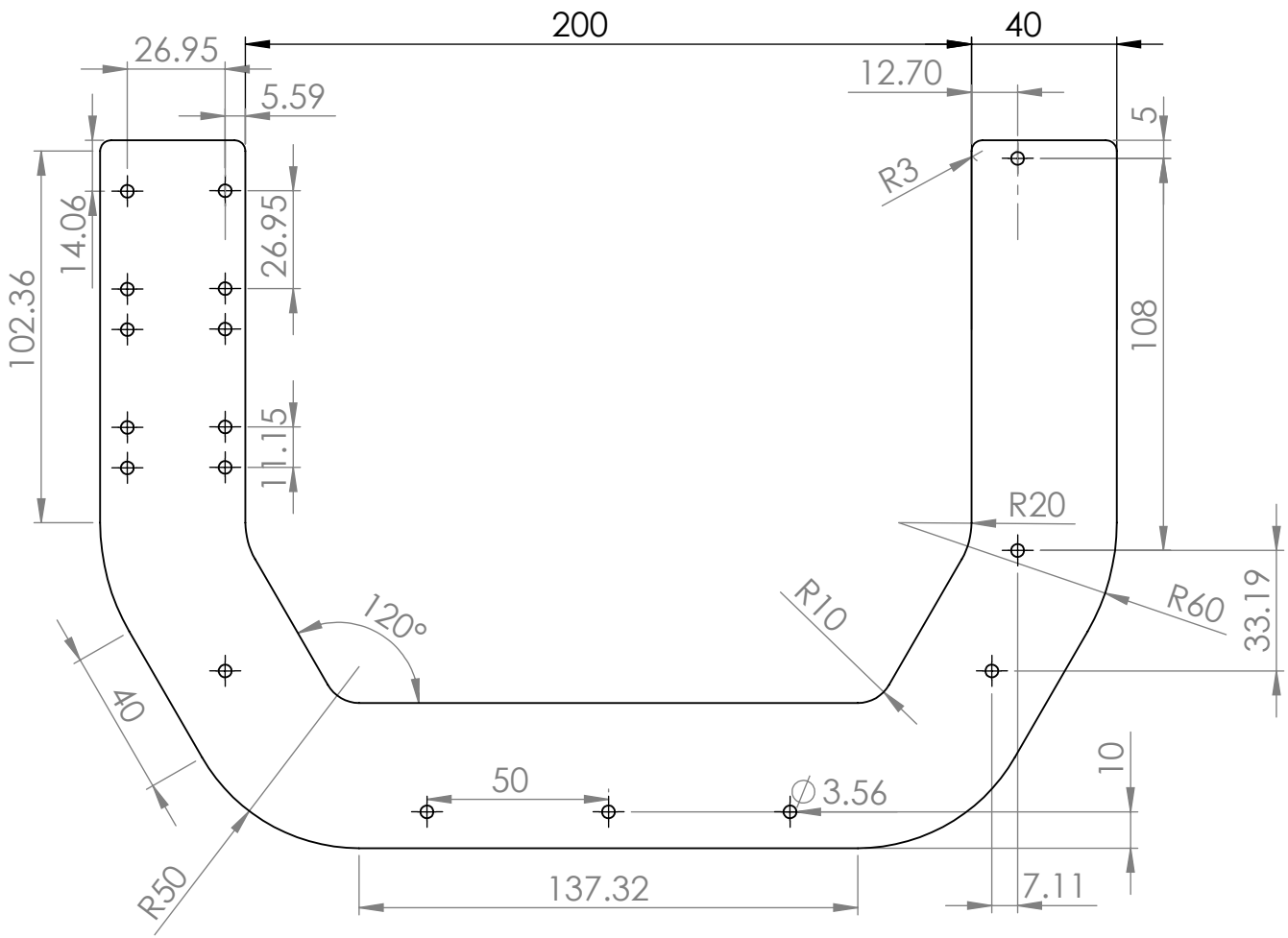


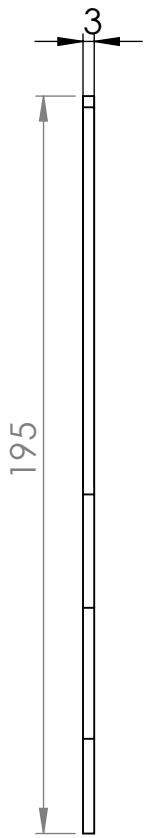


INSTITUTO POLITECNICO NACIONAL		
UNIDAD PROFESIONAL INTERDICIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS		
ALUMNO	GARCIA BARAJAS LUIS ALBERTO	
ESCALA 1:2	DIM. mm.	MATERIAL: ALUMINIO
PROTOTIPO DE SISTEMA PARA SEGUIMIENTO DE OBJETOS CON VISION ARTIFICIAL		
7	BASE LLANTAS	

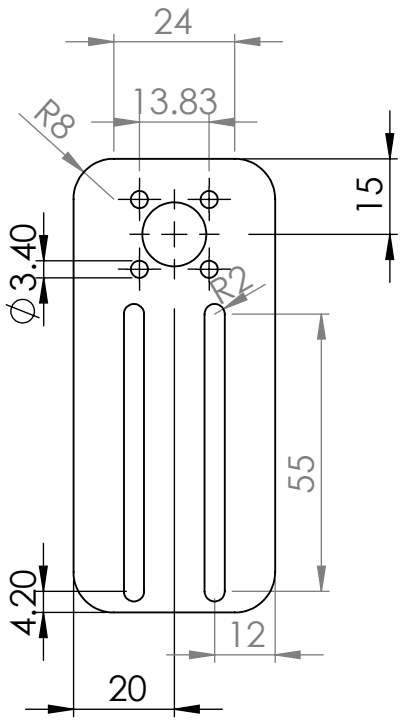


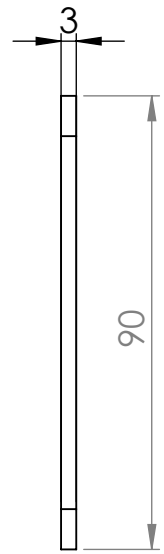
INSTITUTO POLITECNICO NACIONAL		
UNIDAD PROFESIONAL INTERDICIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS		
ALUMNO	GARCIA BARAJAS LUIS ALBERTO	
ESCALA 1:2	DIM. mm.	MATERIAL: ALUMINIO
PROTOTIPO DE SISTEMA PARA SEGUIMIENTO DE OBJETOS CON VISION ARTIFICIAL		
5	BASE RECTANGULO	

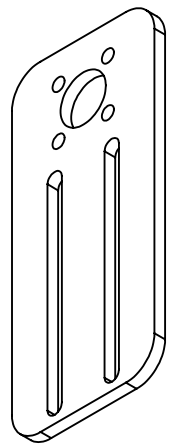




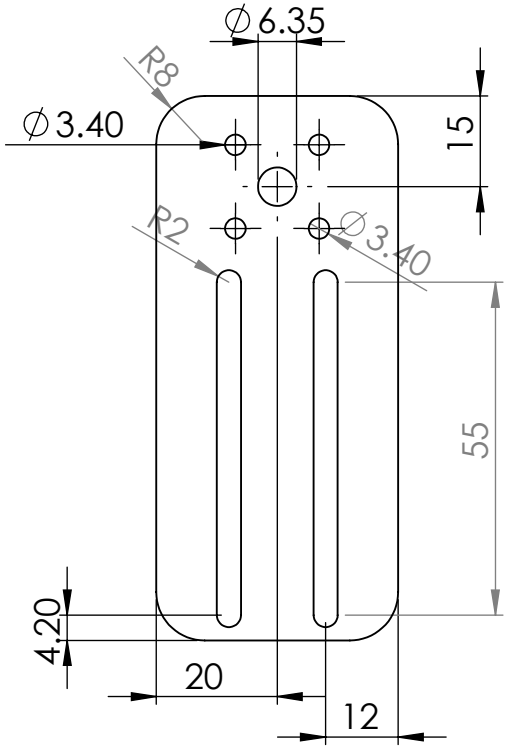
INSTITUTO POLITECNICO NACIONAL		
UNIDAD PROFESIONAL INTERDICIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS		
ALUMNO	GARCIA BARAJAS LUIS ALBERTO	
ESCALA 1:2	DIM. mm.	MATERIAL: ALUMINIO
PROTOTIPO DE SISTEMA PARA SEGUIMIENTO DE OBJETOS CON VISION ARTIFICIAL		
1	ESLABÓN U	

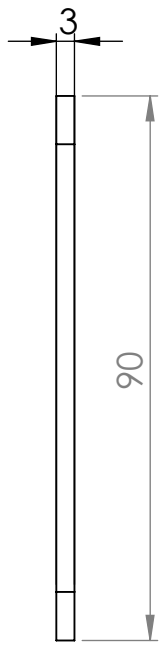


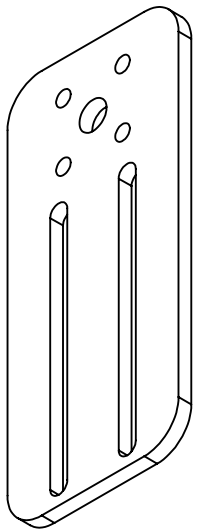




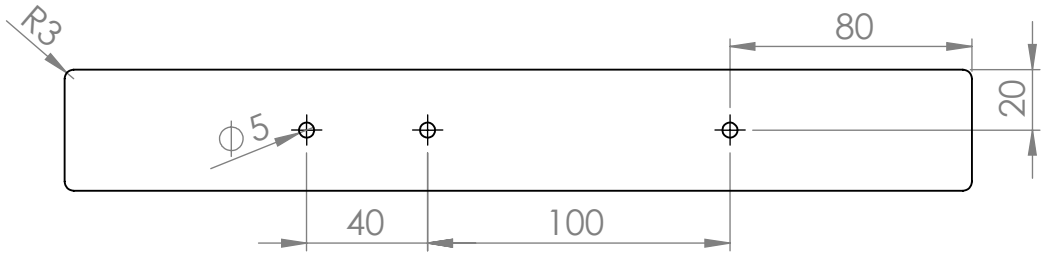
INSTITUTO POLITECNICO NACIONAL		
UNIDAD PROFESIONAL INTERDICIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS		
ALUMNO	GARCIA BARAJAS LUIS ALBERTO	
ESCALA 1:1	DIM. mm.	MATERIAL: ALUMINIO
PROTOTIPO DE SISTEMA PARA SEGUIMIENTO DE OBJETOS CON VISION ARTIFICIAL		
3	PLACA1 CAMARA	

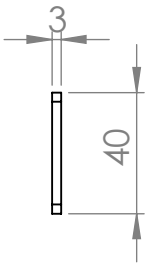


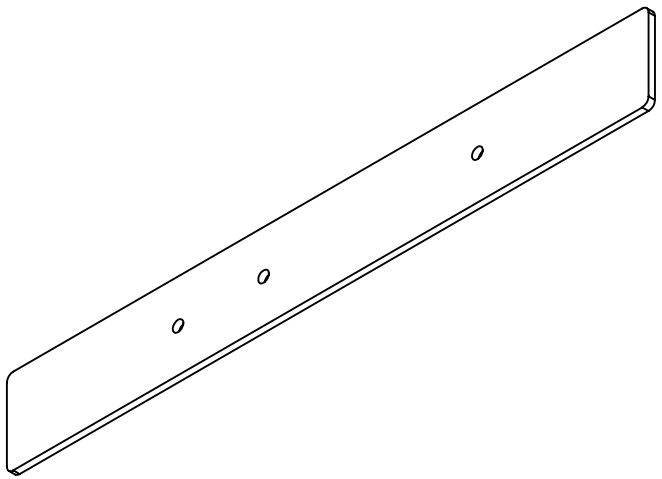




INSTITUTO POLITECNICO NACIONAL		
UNIDAD PROFESIONAL INTERDICIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS		
ALUMNO	GARCIA BARAJAS LUIS ALBERTO	
ESCALA 1:1	DIM. mm.	MATERIAL: ALUMINIO
PROTOTIPO DE SISTEMA PARA SEGUIMIENTO DE OBJETOS CON VISION ARTIFICIAL		
4	PLACA2 CAMARA	







INSTITUTO POLITECNICO NACIONAL		
UNIDAD PROFESIONAL INTERDICIPLINARIA EN INGENIERIA Y TECNOLOGIAS AVANZADAS		
ALUMNO	GARCIA BARAJAS LUIS ALBERTO	
ESCALA 1:2.5	DIM. mm.	MATERIAL: ALUMINIO
PROTOTIPO DE SISTEMA PARA SEGUIMIENTO DE OBJETOS CON VISION ARTIFICIAL		
8	SEPARADOR RIEL	