ADQUISICIÓN DE LA VELOCIDAD Y ACELERACIÓN ANGULAR INSTANTÁNEA DE UN SÓLIDO RÍGIDO MEDIANTE VIDEO

FERNEL AUGUSTO GARCIA BELTRAN
JOSÉ DAVID NIÑO SÁENZ

ESCUELA DE INGENIERIA MECANICA
FACULTAD DE INGENIERIAS FISICO MECANICAS
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA
2015

ADQUISICIÓN DE LA VELOCIDAD Y ACELERACIÓN ANGULAR INSTANTÁNEA DE UN SÓLIDO RÍGIDO MEDIANTE VIDEO

FERNEL AUGUSTO GARCIA BELTRAN
JOSÉ DAVID NIÑO SÁENZ

Trabajo de grado para optar por el título de
INGENIERO MECANICO

Director
JORGE ENRIQUE MENESES FLOREZ
Magister en Ingeniería Mecanica

ESCUELA DE INGENIERIA MECANICA
FACULTAD DE INGENIERIAS FISICO MECANICAS
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA
2015

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

RESUMEN

TITULO: ADQUISICIÓN DE LA VELOCIDAD Y ACELERACIÓN ANGULAR INSTANTÁNEA DE UN SÓLIDO RÍGIDO MEDIANTE UN VIDEO[*].

AUTORES: GARCIA BELTRA Fernel Augusto,
NIÑO SÁENZ José David[**].

PALABRAS CLAVES: Cinemática, adquisición, solido rígido, análisis de video, detección de color, velocidad angular, aceleración angular, fotograma, movimiento plano general, eje de coordenadas, error cuadrático medio

DESCRIPCIÓN:

En la academia, en la asignatura de dinámica se enseñan los conceptos de la cinemática de las partículas y del solido rígido, el estudio y comprensión de los diferentes tipos de movimientos así como la determinación de sus características son fundamentales para la formación del ingeniero mecánico. Sin embargo estos conceptos son teóricos y la correcta percepción del movimiento puede llegar a ser abstracto si se analiza únicamente con las ecuaciones e imágenes presentes en los libros.

Con la finalidad de mejorar el entendimiento de la cinemática se desarrolló una herramienta didáctica con la cual se puede contrastar el movimiento real de un mecanismo con los datos obtenidos de las ecuaciones de la dinámica.

En el presente proyecto se determinan la velocidad y aceleración angular de la biela de un mecanismo biela-manivela a partir de un video del movimiento de dicho mecanismo; para esto se utilizó una herramienta computacional de análisis de video, la cual se modificó para que de manera automática identificará el punto que queríamos analizar dentro del video por medio de detección de marcas visuales de colores predeterminados ubicadas en los puntos de interés del mecanismo al momento de la captura del video. Entonces, con base en este video, se analizaron dos puntos de interés de la biela, obteniendo los datos necesarios para determinar su velocidad y aceleración angular que junto con el sistema de detección de color y el análisis de los resultados obtenidos se exponen de manera detallada a lo largo de la temática de este proyecto.
.

[*] Proyecto de Grado

[**] Facultad de Ingenierías Físico-mecánicas. Escuela de Ingeniería mecánica. Ing. Jorge Enrique Meneses Flórez.

ABSTRACT

TITLE: PROTOTYPE OF A DATA ACQUISICITON SYSTEM TO OBTAIN SURFACE DYNACARDS OF OIL WELLS[*].

AUTHORS: GARCIA LOPEZ Jose Daniel, **
FERREIRA PEREIRA Diego Alejandro[**].

KEY WORDS: Kinematics, data acquisition, rigid body, video analysis, color detection, angular velocity, angular acceleration, frame, general planar motion, coordinate axis, mean squared error

DESCRIPTION:

At the academy, in the subject of dynamic are taught the concepts of kinematics of particles and rigid bodies, the study and understanding of the different types of movements and determine their characteristics are essential for the formation of mechanical engineer. However, these concepts are theoretical and the correct perception of motion can be abstract if analyzed only with equations and images on the books.

In order to improve understanding of the kinematics it developed an educational tool to contrast the real movement of a mechanism with the data obtained from the equations of dynamics.

In this project the velocity and angular acceleration of the rod of a rod-crank system are determined from a video of the motion of this mechanism; A computational tool for video analysis was used, the tool was modified to automatically identify the point we wanted to analyze within the video through visual detection of marks with default colors located at points of interest when the video of the mechanism was recorded. Then, based on this video, two points of the rod were analyzed, obtaining the necessary data to determine its velocity and angular acceleration with the system color detection and analysis of the results obtained are set out in detail along the theme of this project.

[*] Graduation Project

[**] Faculty of Physical – Mechanical Engineering. Mechanical Engineering School. Eng. Jorge Enrique Meneses Florez.

INTRODUCCIÓN

La Universidad Industrial de Santander y en este caso, la escuela de Ingeniería Mecánica han inculcado en sus estudiantes el uso de diversas herramientas para tener una formación integral durante su proceso de aprendizaje.

En los últimos años el uso de software ha hecho posible tener herramientas para un mejor aprendizaje de las asignaturas vistas en la academia y así de manera didáctica contrastar la realidad con la simulación por computadora para de esta forma asimilar de manera más clara los conceptos; dos de estas herramientas son el análisis de video y el modelado de sistemas dinámicos.

De esta manera, hoy en día, gracias a los avances de la tecnología y la capacidad de los ordenadores podemos encontrar una gran cantidad de software de computadora que buscan simular los diversos temas de asignaturas como Mecánica, Resistencia de materiales, Dinámica y otras más; software utilizados para determinar las características físicas de problemas reales. Específicamente en el campo del estudio del movimiento, la simulación por computadora permite a los estudiantes poner un modelo dinámico de partículas en un clip de video, donde se determina una escala para el video y se definen de forma apropiada los ejes de coordenadas para el análisis del video y de esta forma poder comparar videos del mundo real con animaciones de los modelos teóricos. Un video es sencillamente un conjunto de imágenes o fotogramas mostrados de manera secuencial, entonces la expresión “análisis de video” hace referencia al análisis de cada una de los fotogramas que componen el video.

Este trabajo de grado se enfocó en analizar el movimiento plano general del solido rígido, el cual es la adición del movimiento de traslación combinado con el

movimiento rotacional; además, se buscó adquirir los datos de velocidad angular y aceleración angular de dicho solido rígido y además incorporar detección de marcas para poder utilizar de manera más intuitiva el software.

El presente libro explica de manera detallada los temas mencionados anteriormente. Los capítulos se organizaron con el objetivo de lograr una mayor comprensión del trabajo realizado de la siguiente manera:

1. Importancia y objetivos del proyecto. En esta parte del libro se incluyen el objetivo general y los objetivos específicos del proyecto, así como el planteamiento del problema y la justificación del mismo.

2. Adquisición de la velocidad angular y aceleración angular por medio de video. Presenta la idea general del proyecto mostrando los diversos desarrollos realizados a lo largo de la ejecución del presente proyecto de una manera resumida.

3. Obtención de imágenes. Presenta los elementos usados en el desarrollo de la obtención del primer fotograma del video, también de los elementos que participaron en la creación del video y las diferentes pruebas que se hicieron.

4. Módulo Ident-object (detección automática de puntos y regiones). Aquí se explica el modo en que se desarrolló la identificación automática de los marcadores que permiten la automatización del software.

5. *Módulo TRACKER*. En este capítulo se hace un análisis de los diferentes elementos por los que está compuesto TRACKER, como es su operación y el algoritmo que usa al momento de realizar el análisis de los datos generados en el estudio cinemático.

6. Automatización del software TRACKER. Presenta el desarrollo de investigación y cambios que se realizaron en el software de estudio de video TRACKER, explicando paso a paso el código que permitió obtener el producto final.

7. Análisis cinemático. Es el capítulo donde se muestra las ecuaciones cinemáticas y su estudio del mecanismo de prueba del software de video, que en este caso es un mecanismo manivela- biela- corredera.

8. Análisis de resultados. Esta sección muestra los resultados obtenidos a partir de la obtención de datos del análisis cinemático hasta el momento en que se encuentra la velocidad angular y aceleración angular del solido rígido.

9. Conclusiones. En este capítulo podemos identificar y concluir si los objetivos planteados fueron cumplidos y el análisis del proyecto en general.

10. Recomendaciones.

11. Anexos. En esta sección se tiene toda la información complementaria referente al proyecto como lo es marco teórico, hojas de especificaciones, planos y demás.

1. IMPORTANCIA Y OBJETIVOS DEL PROYECTO

1.1 IDENTIFICACIÓN DEL PROBLEMA

Actualmente en la escuela de Ingeniería Mecánica de la Universidad Industrial de Santander se encuentran cursando aproximadamente 90 estudiantes la asignatura de DINÁMICA, la cual es una materia en la que los estudiantes afianzan conocimientos elementales que les servirán para el desarrollo y comprensión de otras materias futuras, siendo esta requisito de asignaturas fundamentales como TERMODINAMICA I, MECANICA DE FLUIDOS y MECANICA DE MAQUINAS. La metodología de esta materia actualmente es basada en la teoría, suministrando la información en las aulas de clase con las horas catedra dictadas por los profesores y consultando la teoría y resolviendo ejercicios de los libros en horas de trabajo en casa.

La dinámica se evidencia más allá de los esquemas de los libros, en lo cotidiano, es posible observar pateando un balón el movimiento parabólico de un proyectil, o en la colisión de dos bolas durante un juego de billar podemos ver aplicados los principios de la teoría de colisiones e incluso en nuestra formación como ingenieros es importante comprender los diferentes movimientos de las maquinas; sin embargo, nuestros estudios no tienen ningún campo practico. Vemos que los estudiantes que ven la materia de dinámica del cuerpo rígido no poseen un laboratorio donde pongan en práctica lo aprendido en las aulas de clase, de todos los laboratorios utilizados por esta escuela ninguno es enfocado al desarrollo de esta materia. Lo cual hace muy dificultoso entender y percibir algunos tipos de movimientos.

1.2 JUSTIFICACIÓN PARA SOLUCIONAR EL PROBLEMA

La Universidad Industrial de Santander busca formar a los estudiantes de forma integral, en su búsqueda por el mejoramiento metodológico donde el estudiante logre asimilar más fácilmente la información suministrada por los profesores, se desea beneficiar en la asignatura de dinámica con una metodología de aprendizaje práctica, además de la teórica suministrada hasta el momento. Ya que se debe tener en cuenta que existen diferentes formas de adquirir conocimientos para los estudiantes, mientras para algunos es fácilmente entendible y asequible por medio de una clase cátedra para otros no lo puede ser tanto, este tipo de estudiantes por lo general, necesitan adquirir el conocimiento de una forma más lúdica, donde se pueda afianzar el conocimiento dado en las horas cátedra y así lograr un complemento que ayuda a la formación del estudiante como ingeniero.

Tomando este caso para la asignatura de dinámica se propuso tener un software donde sea posible mediante el video de un movimiento cinemático de un sólido rígido en dos dimensiones, determinar las características como velocidad y aceleración angular, para así colocar en contraste los resultados reales y los resultados encontrados en el papel, para corroborar que el procedimiento teórico utilizado haya sido el correcto y además de la percepción espacial del movimiento que podrá desarrollar el estudiante, al observar el movimiento real de lo que se dispone a calcular.

Este trabajo de grado contribuye a la percepción y entendimiento por medio práctico de un aspecto importante de la dinámica como la cinemática del cuerpo rígido a los estudiantes de esta asignatura, se espera que sea una herramienta fácil de utilizar y con proceso operativo sencillo y predeterminado.

1.3 OBJETIVOS DEL TRABAJO DE GRADO

1.3.1. Objetivos generales

- Contribuir con el propósito de la universidad, ayudando a los estudiantes de ingeniería mecánica en el área de la dinámica, a desarrollar mayor comprensión y competencia en los temas abordados en esta asignatura.

1.3.2 Objetivos Específicos

- Diseñar y desarrollar un software de modelamiento computacional de video, mediante el cual se pueda analizar el movimiento de un sólido rígido y que posea las siguientes características:
  - Que permita obtener las características cinemáticas instantáneas del movimiento plano general de un sólido rígido en dos dimensiones, tales como la posición, velocidad y aceleración angular.
  - Utilice identificadores visuales previamente dispuestos en el sólido rígido en movimiento.
  - Permita analizar de forma simultánea dos (2) elementos del sólido rígido en movimiento.
  - Software en lenguaje java
  - El video a analizar debe tener máximo 20 cuadros/segundo.
  - Los formatos de video soportados se limitan a mov/ avi/ flv/ mp4

- Diseñar el procedimiento de la creación del vídeo del movimiento del solido rígido, para el correcto análisis cinemático por parte del software y así obtener los resultados esperados.

2. SINTESIS: ADQUISICION DE LA VELOCIDAD Y ACELERACION ANGULAR INSTANTANEA DE UN SOLIDO RIGIDO MEDIANTE VIDEO

El presente capitulo expone la idea general del proyecto, mostrando los avances realizados a lo largo de la ejecución del mismo, de una manera resumida.

2.1 INTRODUCCIÓN

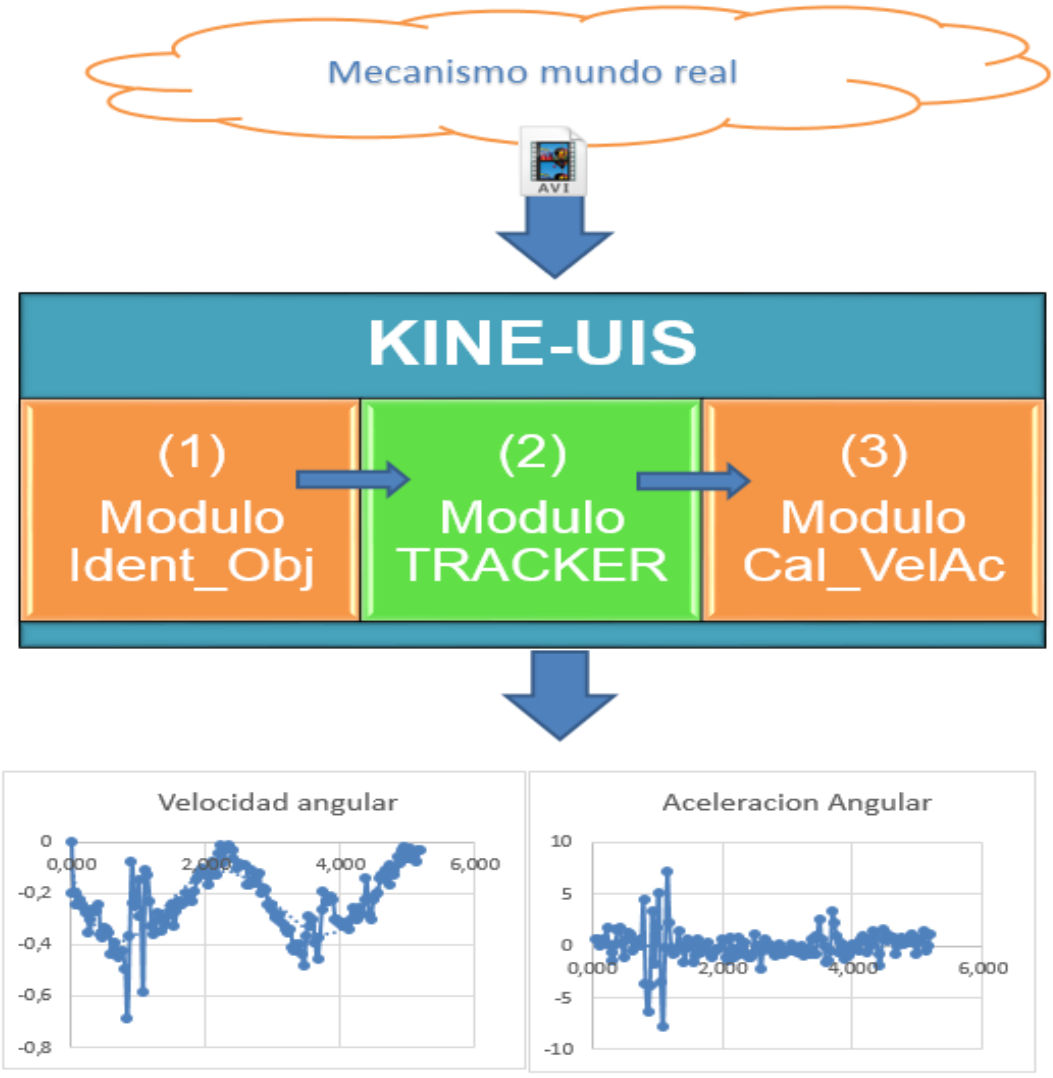
El desarrollo del proyecto surge del interés de estudiar y obtener las características cinemáticas de los movimientos de mecanismos que encontramos en la vida real; para poder analizar este tipo de mecanismos se emplea un video del movimiento, el cual se le suministra a un software que analiza las imágenes del video para obtener los datos cinemáticos del movimiento, la idea general puede verse en la figura 1.

Figura 1. Diagrama general del proyecto.

2.2. KINE-UIS

Para lograr obtener a partir de un video las características cinemáticas de un sólido rígido en movimiento, se desarrolló un software que se ha denominado "*Kine-UIS*". El desarrollo de *Kine-UIS* se basó en "TRACKER", software de uso libre programado en lenguaje java. En la figura 2 se muestra el esquema general del software *Kine-uis*

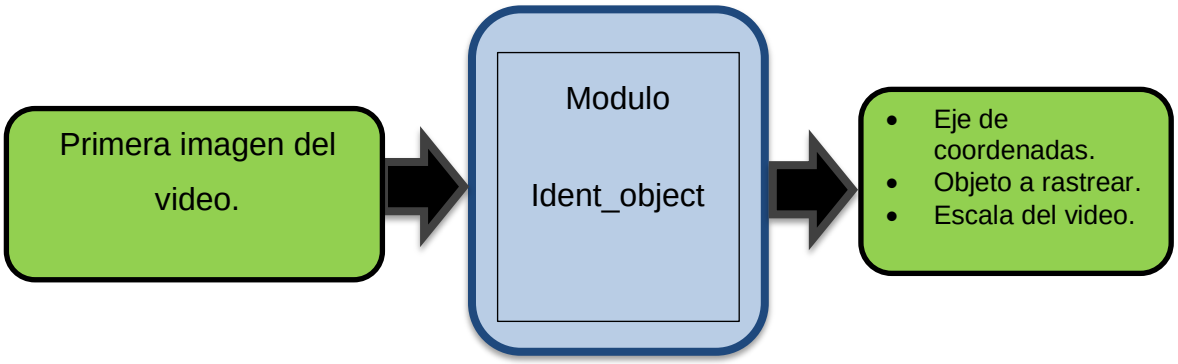
Figura 2. Diagrama KINE-UIS.



Kine-UIS está compuesto por 3 módulos principales, su entrada es el video del mecanismo y la salida final son los datos de velocidad angular y aceleración angular con respecto al tiempo. A continuación se describen cada uno de los módulos desarrollados.

2.2.1 Modulo Ident-Obj.

Figura 3. Diagrama del módulo Ident-object.

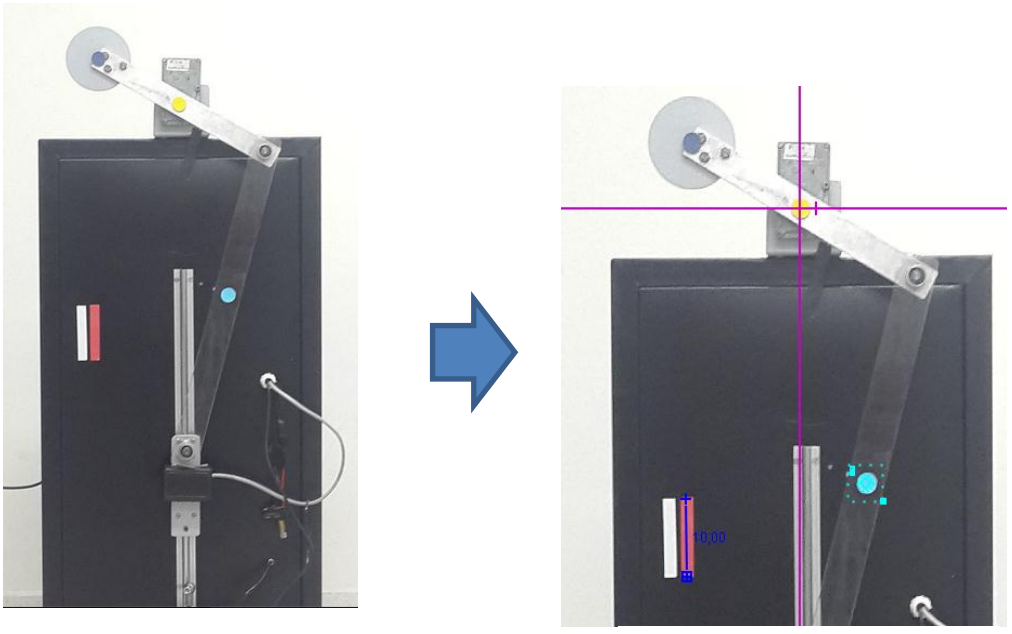


Al abordar cualquier ejercicio de cinemática hay ciertos parámetros que deben definirse inicialmente, estos son: el origen del sistema de coordenadas, las magnitudes de las distancias y el objeto el cual se desea examinar; sin embargo, en la vida real no se tienen definidos estos datos y al grabar el video de un mecanismo, las imágenes del video en la realidad no asumen por si solas un eje de coordenadas, tampoco se conocen las distancias de los objetos de la imagen, ni se tiene claro que objeto se analizará, por ello, se desarrolló este primer módulo de *Kine-UIS*.

El módulo *Ident Obj* se encarga de definir los 3 parámetros nombrados anteriormente, para lograr esto, se incorporó en este módulo la librería de visión artificial “OpenCV”, con la cual *Kine-UIS* detecta de manera automática, antes de comenzar el análisis, tres marcas de colores que se ubican en el mecanismo al momento de grabar el video, de esta manera, el usuario con simplemente colocar

estas marcas ya habrá definido donde quiere situar el origen del eje de coordenadas, ya tendrá cual es la escala que se utilizará para hacer los cálculos y además se tendrá el punto el cual se desea analizar; en la ilustración 04 se muestra el primer fotograma de un video del mecanismo de prueba, y se observa la detección del circulo azul claro como el punto a analizar, el circulo amarillo como el origen del sistema de coordenadas y la franja vertical roja como la escala del video. La forma como se realizó la detección de colores se explica de manera amplia en el capítulo 4.

Figura 4. Identificación de marcas.



(a)

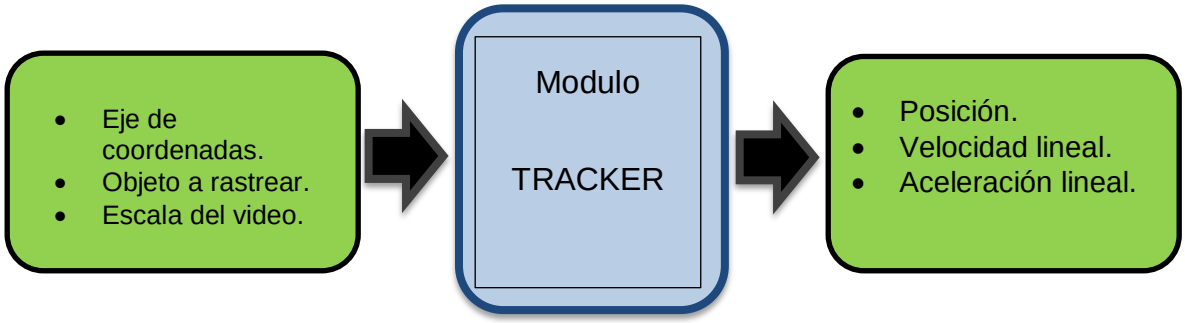
Sin identificar marcas.

(b)

Marcas identificadas.

2.2.2 Modulo TRACKER.

Figura 5. Diagrama del módulo TRACKER.



Para el análisis de video, *Kine-UIS* emplea "TRACKER" la cual es una herramienta de modelado y análisis de video, software hecho en java, que utiliza la librería "Open source physics" y que ha sido diseñado con fines educativos en el área de dinámica.

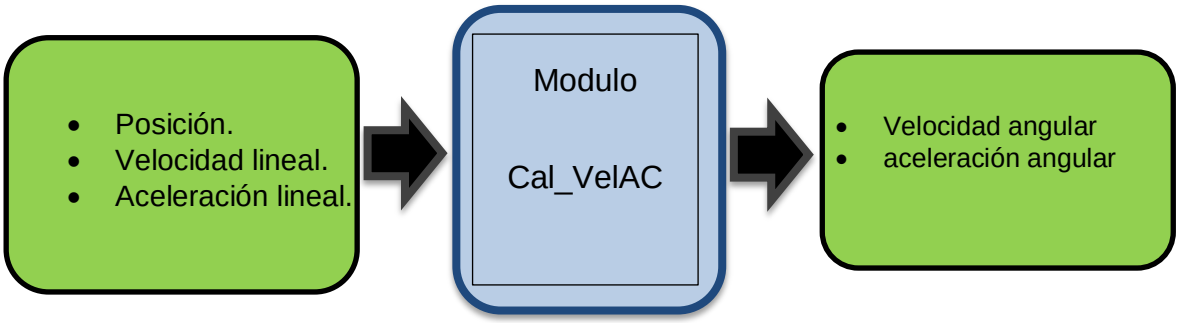
El trabajo realizado en este módulo, es el acoplamiento de la identificación automática de colores realizado en el módulo *Ident_Obj* con TRACKER, esto quiere decir que se modificó el código de TRACKER para que recibiera los parámetros de entrada definidos en el módulo anterior para comenzar con el análisis del video. En el capítulo 6 se muestran las partes del código que tuvieron que ser modificadas junto con los cambios que se realizaron en el código original.
Con los datos del módulo *Ident_Obj*, el módulo *TRACKER* procede a hacer el "rastreo", durante el cual recorre cada uno de los fotogramas del video marcando en ellos la posición central del objeto de análisis definido, de esta manera, después de concluir el análisis de todo el video, el software guarda un registro de datos de las coordenadas de los pixeles que se marcaron en cada fotograma del video con su respectivo instante de tiempo, y utilizando la escala tenemos así mismo los datos de posición respecto al tiempo de el objeto predeterminado; cabe aclarar que el TRACKER está limitado a analizar objetos puntuales o partículas,

aunque es posible analizar varias objetos puntuales del mismo video, sin embargo el software desarrollado en este proyecto, analiza la cinemática de un cuerpo rígido en movimiento plano general.

2.2.3 Modulo Cal-VelAc. Debido a que TRACKER analiza únicamente objetos puntuales, y se desea analizar un movimiento plano general de un sólido rígido, se desarrolló este módulo final en *Kine-UIS*, en el cual con los datos obtenidos de TRACKER, se calcula la velocidad y la aceleración angular de un sólido rígido, utilizando las ecuaciones de cinemática mostradas en el capítulo 7.

El trabajo realizado además del cálculo de la velocidad y aceleración angular también incluye el análisis de los resultados, en el cual se verifica la fidelidad de los datos obtenidos y en los casos de errores altos se filtraron los datos para reducir el error debido al ruido de las señales.

Figura 6. Diagrama del módulo *Cal-VelAC*.



El proceso de filtrado se realizó exportando los datos a MatLab y aplicando una función que contiene MatLab cuyo nombre es Smooth que consiste en un filtro pasa bajas, que permite suavizar la curva generada, la cual fue aplicada para los datos de velocidad angular, para los de velocidad, aceleración y aceleración angular permitiendo así una reducción en el ruido que llegaban a los datos por razones de error a la hora de tomar el video como el desbalanceo, errores agregados por el uso de ecuaciones sucesivas y el error al tomar datos discretos.

3. OBTENCIÓN DE IMAGENES.

3.1. VIDEO

En el momento de hacer la adquisición del video se hizo por medio de la cámara de un celular Samsung S4 que tiene estabilizador de imagen, enfoque automático, grabación de videos en HD el cual trabaja a 30 fps (fotogramas por segundo), se debía tener la precaución de no tomar en ángulo inclinado el video, en vez de eso grabarlo de frente al banco de pruebas, por lo que se procedió a instalar un trípode con medidor de nivel, el cual aseguró tener una adecuada forma de realizar la filmación del video o adquisición de fotogramas, ya que si se hacía de un ángulo incorrecto ocasionaría errores al analizar el video en *Kine-UIS*, por lo que se debe buscar que el movimiento que se genere en el mecanismo sea lo más cercano a un movimiento de dos dimensiones y TRACKER no cuenta con la capacidad de hacer un estudio cinemático donde esté presente un eje adicional, esto es en tres dimensiones, que en este caso sería la profundidad, es por esto que se dispuso el banco en paralelo con la cámara.

Es importante la detección de diferentes colores en el video, por lo tanto se hicieron pruebas de luces durante la filmación del mecanismo operando, para observar que tipo de iluminación al momento de tomar el video influye mejor en el software cuando comienza a realizar el análisis, ya que se debía garantizar que durante el video los indicadores de colores fueran visibles, ya que esto podría producir una reflexión de luz y hacer que el software al momento de realizar el estudio cinemático se pueda perder al intentar encontrar la masa puntual, también se aseguró que no se generaran sombras. Además de hacer un adecuado enfoque y limpieza del escenario, para asegurar que no existieran objetos que contengan los mismos colores de los indicadores.

Cuando se analizaron videos que contenían pequeños elementos del mismo color de alguno de los marcadores que se pretendía detectar, el módulo de detección presentaba fallos debido a que los pixeles del mismo color son identificados por el algoritmo, entonces no se puede asegurar que la marca de color que se ubicó inicialmente sea la que detecte el software y existe posibilidad que detecte primero el conjunto de pixeles del otro objeto que se encuentre de igual forma en el video.

3.2. MECANISMO

Las pruebas que se realizaron del software TRACKER para la adquisición de la velocidad angular y aceleración angular fueron tomadas en el laboratorio de automatización industrial, aplicándolo a un mecanismo manivela – biela – corredera, el cual fue tomado de un anterior proyecto titulado "PROTOTIPO DE UN SISTEMA DE ADQUISICION DE DATOS PARA LA OBTENCION DE DINAGRAMAS DE SUPERFICIE DE UN POZO PETROLERO"[1], cuyos autores fueron Diego Alejandro Ferreira y José Daniel Garcia bajo la dirección del ingeniero mecánico Jorge Enrique Meneses Flórez, este mecanismo cuenta con la posibilidad de conectarse a un PLC SIEMENS, además tiene un amplificador de instrumentación, cuenta además con un relé del electroimán y el relé del motor el cual es el que se usó para la alimentación y puesta en marcha del motor, la disposición de este mecanismo es de forma en que el carretel se mueve de forma que sigue una guía vertical.

El motor se alimenta de un voltaje de 24v el cual esta acoplado a una caja de reducción la cual es la que le da la salida al eje que mueve la manivela, la manivela tiene una masa inercial en uno de sus extremos, la cual actúa como

[1] FERREIRA, Diego Alejandro y GARCIA, José Daniel. Prototipo de un sistema de adquisición de datos para la obtención de dinagramas de superficie de un pozo petrolero. Trabajo de grado Ingeniero Mecánico. Bucaramanga.: Universidad Industrial de Santander. Facultad de Ingenierías Físico Mecánicas. Escuela de Ingeniería Mecánica, 2012. 169 p.

contrapeso en los diferentes momentos de trabajo del motor, si no estuviera ahí esa masa inercial, la cantidad de torque que ejerce el motor al elevar la carga sería mucho mayor que el torque necesario para hacer que la carga baje, tomando como carga el peso de las barras que componen el mecanismo.

Las barras que componen el mecanismo son hechas de aluminio, las cuales se encuentran acopladas mediante pequeños ejes, los cuales son sostenidos por rodamientos; el banco tuvo que ser adecuado para este proyecto, puesto que es necesario que cada uno de los puntos de las barras sean visibles durante los videos, se tomó la decisión de cambiar la biela de aluminio por una de acrílico (transparente) de la misma medida, ya que se observó que al momento de la puesta en marcha del mecanismo por momentos se genera una superposición entre la manivela y la biela lo que imposibilita a la cámara seguir y encontrar los identificadores puestos en diferentes puntos del mecanismo, por lo que al momento de incluir la biela de acrílico era posible ver correctamente el indicador de la masa puntual en cada instante de tiempo que dura el video.

Figura 7. Biela de acrílico y banco de pruebas del mecanismo manivela-biela-corredera.

3.3. FUENTE

La fuente de alimentación del motor, es una fuente de alimentación de tres canales, pero para hacer funcionar el mecanismo, solo es necesario usar un canal. Se debe considerar de que el mecanismo contiene una masa que hace que el esfuerzo necesario para moverlo difiera al momento de hacer que el mecanismo lo suba o lo baje, es por esto que al momento de consumo de corriente desde la fuente al mecanismo esta varié, ya que necesita mayor Intensidad de corriente [A] para que el mecanismo logre subir el peso de las barras del mecanismo.

Figura 8. Fuente de poder.

Fuente: Autores.

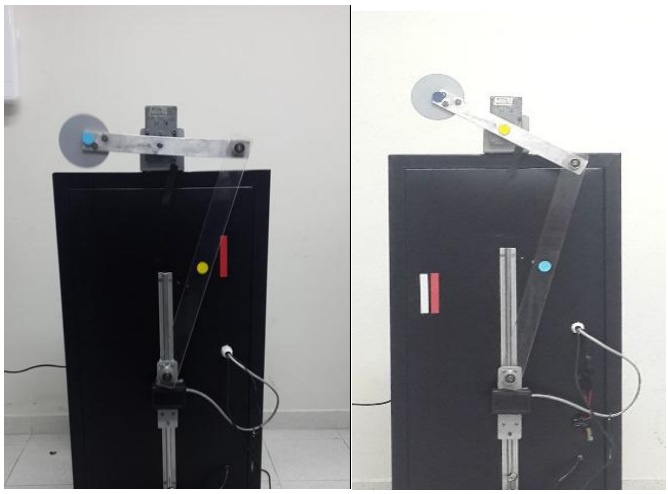
3.4. VARIACION EN LA POSICION DE LOS INDICADORES

Se hicieron diferentes pruebas para determinar correctamente la mejor relación de los indicadores, con el objetivo de asumir cual color es la mejor elección para cada indicador, esto es necesario para el software al momento de realizar el análisis cinemático. En un principio se dispuso que la barra de calibración fuera amarilla, el

indicador de la masa puntual fuera roja y el eje de coordenadas fuera azul. Esta disposición de indicadores tenía un excelente comportamiento, si el video era tomado correctamente, todos los identificadores aparecían en el lugar correcto luego de haber hecho clic en el icono creado AUTOTRACKER.

Pero se decidió hacer pruebas con otro tipo de configuraciones al momento de elegir el color adecuado para cada indicador, esta vez se eligió amarillo como eje de coordenadas, rojo como barra de calibración y azul como masa puntual. Estos cambios son hechos en el código realizado en ECLIPSE, por lo que es un proceso complejo el cual lleva tiempo. Al realizar los cambios y poner en marcha de nuevo el software se demostró que tenía un correcto funcionamiento por lo que se concluyó que la configuración de los colores al momento de caracterizar cada indicador no era importante ya que servía de forma adecuada sin importar la configuración de colores usada, es decir cumplía con el objetivo planteado.

Figura 9. Diferencia de colores en los identificadores.



Por lo que se dejó la última configuración que se tomó, ya que volver a realizar los cambios como se comentó anteriormente es un proceso dispendioso y largo, por lo cual se determinó que era innecesario. La disposición finalmente que se tomó fue:

Masa Puntual = Color AZUL CLARO

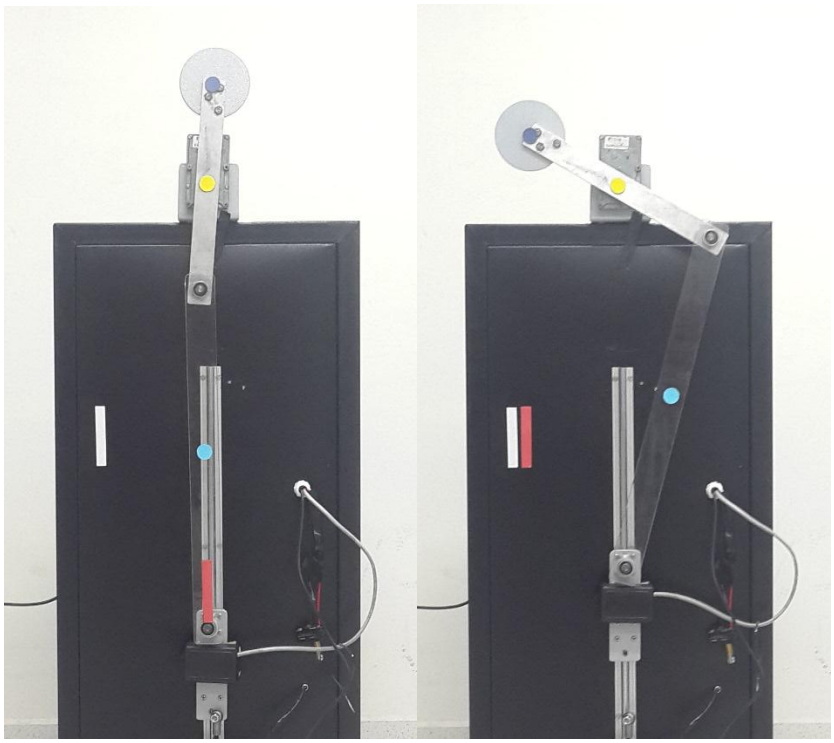
Barra de Calibración = Color ROJO

Eje de Coordenadas = Color AMARILLO

Figura 10. Barra de calibración horizontal.

También se cambió la posición de la barra de calibración y se encontró que el lugar óptimo para ponerlo es un lugar estático, y no uno en movimiento, además de hacer la prueba, con la barra en forma horizontal, la cual no funciono ya que el código que se agregó sirve solo para identificar la barra de forma vertical y no horizontal, por lo que al hacer el estudio con un video con la barra horizontal no se lograba determinar de forma adecuada la escala del video.

Figura 11. Variación en la posición de la barra de calibración.



3.5. SIMULACION EN VARIOS PUNTOS

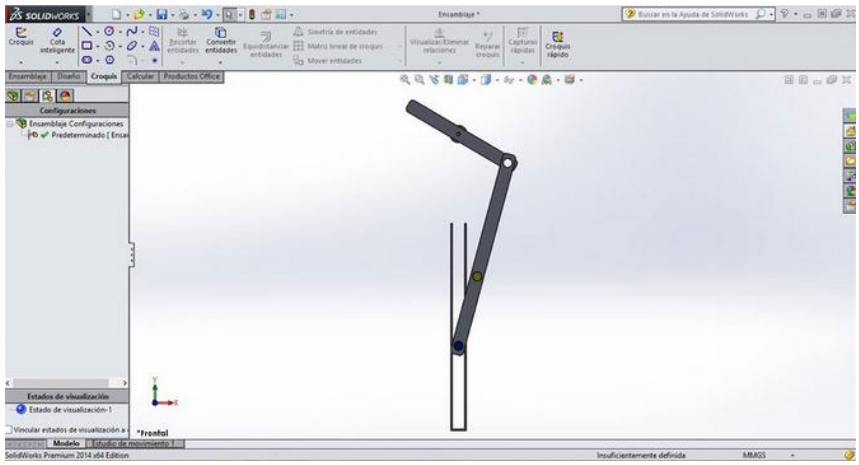
Para la determinación de la velocidad angular y la aceleración angular fue indispensable realizar el estudio cinemático de video mediante TRACKER, efectuando varias simulaciones. Debido a que es un mecanismo manivela- biela – corredera, se realizaron tres simulaciones, para lo cual se grabaron tres videos diferentes, colocando el indicador de la masa puntual en cada uno de los puntos de interés, una de ellas en el punto donde se encuentra el punto que conecta la manivela y la biela, otra en un punto medio de la biela y la tercera en la corredera, estos tres puntos son utilizados para el cálculo por medio de ecuaciones para determinar la velocidad y aceleración angular.

Estos datos fueron trabajados en TRACKER, al momento de usar las tablas de las diferentes simulaciones a la vez fue necesario exportar todos los datos a una hoja de datos de Excel para poder utilizarlas al mismo tiempo.

3.6. SIMULACION EN SOLIDWORKS

Debido a que la simulación del mecanismo real no presentaba una velocidad angular constante en la manivela, se vio la necesidad de hacer pruebas con un mecanismo con velocidad angular constante en la manivela, por lo que se decidió crear en Solid Works una simulación de un mecanismo manivela – biela – corredera, con una velocidad constante en la manivela, estos se encuentran con las mismas dimensiones que el mecanismo real.

Figura 12. Mecanismo simulado en Solid Works.



Con este mecanismo con velocidad angular constante se procedió a realizar una simulación para determinar si los datos que en ella se generaban y la adquisición de la velocidad angular y la aceleración angular en cualquier punto de la biela son correctos, comparándolos con los datos que se obtuvieron del mecanismo real,

teniendo la seguridad que el método de hallar la velocidad y la aceleración angular mediante el software TRACKER es correcto.

4. MODULO IDENT_OBJECT (DETECCION AUTOMATICA DE PUNTOS Y REGIONES)

Para la detección automática de los puntos y líneas de referencia que son necesarios para el software TRACKER a realizar los cálculos de posición, velocidad y aceleración lineal se debe hacer primero una identificación de estos, es posible hacerlo con un diagrama de colores que permita identificar un color con un punto de referencia determinado, por ejemplo, que el amarillo corresponda al punto referencia del eje coordenado, y así sucesivamente con la distancia de calibración y la masa a analizar, esto es posible gracias a la librería de OpenCV.

Vale aclarar que para que se produzca un adecuado reconocimiento de colores por parte del software se debe ser preciso con los colores que se van a utilizar y no tener más objetos del mismo color en el plano del video que pueden llegar a hacer que el software se confunda con los objetos a analizar, por ejemplo si la masa a analizar se identifica con el color rojo, no debe haber más objetos de este color en el video aparte de la masa que se desea analizar, y la otra forma de ayudar al software es colocando un fondo que contraste completamente con los colores que se van a analizar.

Para eso se debe entender cómo funciona correctamente la clasificación de los colores por lo tanto se explicarán los modelos de colores utilizados.

4.1. MODELOS DE LOS COLORES

4.1.1. Modelo RGB RGB es un modelo, el cual da su nombre por sus siglas en inglés (red, Green y blue), el cual indica que un color está compuesto por la

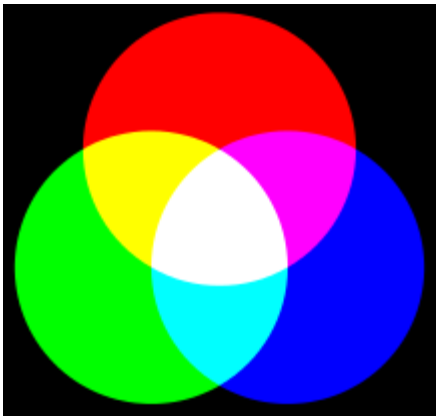
combinación en proporción de estos tres colores, este es un modelo altamente utilizado, es capaz de identificarse con este modelo 16,7 millones de colores.

Este modelo lo que hace es dar en un pixel este color, pero este pixel se divide en 3 sub-pixeles, los cuales están compuestos por estos tres colores, existen dos formas de denominarlo, una hexadecimal y otra decimal, en la decimal el grado de cada color oscila entre los valores de 0 y 255, donde el 0 significa ausencia de ese color y 255 significa que el color a analizar está compuesto por ese color en su totalidad, su sintaxis es (R, G, B). Por ejemplo el color rojo es (255, 0, 0), el verde es (0, 255, 0) y azul (0, 0, 255), otro ejemplo sería el blanco, que de acuerdo a la definición es la suma de todos los colores (255, 255, 255) y el negro la ausencia de todos (0, 0, 0).

Y la notación hexadecimal está compuesta por tres pares, cada par le corresponde a uno de los tres colores antes mencionados, rojo, verde y azul. Los valores oscilan entre 0 y 9 siendo 0 el mínimo y 9 el máximo o también varía entre A y F siendo A el mínimo valor y F el máximo, como ejemplo usaremos el color blanco (#FFFFFF) y negro (#000000), rojo (#990000), verde (#00FF00) y azul (#000099).

Como este modelo es uno de los más utilizados en el mundo nos pareció conveniente usarlo en la identificación del color en el software al momento de identificar el objeto, pero al momento de trabajar con él se complicaba bastante su funcionamiento ya que se debían tener en cuenta tres datos de tres colores diferentes y entrar a comparar cada uno con el del color a detectar, entonces como los cuadros que saldrán durante el recorrido del video podrían llegar a contener varios colores, y ahí es donde el software debe mostrar que es capaz de identificar el correcto, por tanto se vio la necesidad de implementar otro modelo de color que nos permita identificar el color con solo un nivel de identificación de color y esto se hizo convirtiendo el RGB a LAB.

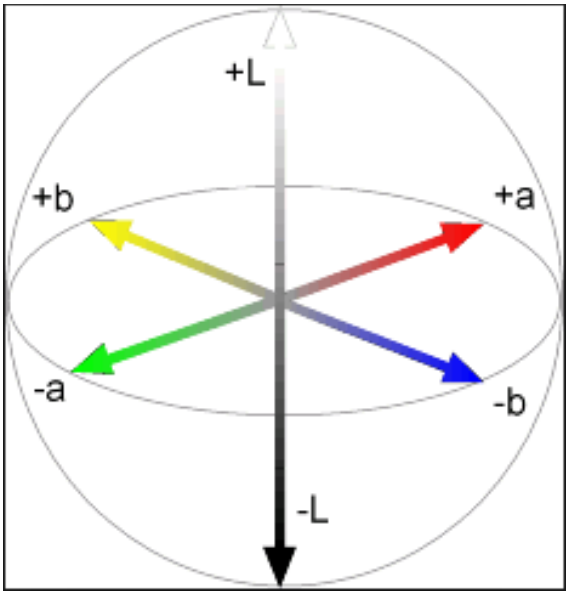
Figura 13. Modelo aditivo de colores RGB.



Fuente: Sebastien-notet

4.1.2. Modelo LAB Este modelo también es altamente utilizado, ya que identifica todos los colores posibles de captar por el ojo humano, este modelo de colores está compuesto por tres componentes L, A y B, para lo cual la L significa la luminosidad, a partir de esta es que se basa el modelo, la A va del color rojo a verde, siendo verde el valor negativo y rojo el positivo y B va de color azul a amarillo, donde los valores negativos son azul y el amarillo positivos, el modelo está representado en esta gráfica.

Figura 14. Espacio de color LAB.



Fuente: Sobrecolores.

4.1.3. Elección del modelo de color indicado. Se decidió tomar el modelo de color LAB para trabajar en el proyecto de grado debido a que en RGB era necesario tomar tres valores y trabajar con cada uno de ellos, en cambio en LAB solo era necesario saber el grado de luminosidad y en que parte de las líneas de color se analizarían, por ejemplo si se decidía detectar el color rojo solo debíamos mirar el lado positivo de la A, esto hace un poco más fácil el trabajar en el reconocimiento de los colores, también hay que tener en cuenta que la librería con el cual se desarrolló el software trabaja bien estos dos modelos de colores.

OpenCV hace la conversión automática entre estos dos diferentes modelos de colores los cuales son guardados en unas matrices en el software para luego trabajar con ellos y efectuar la comparación, simplemente se debe colocar la librería de OpenCV, la cual es **cv,** por tanto se debe leer primero la imagen, se debe convertir la imagen debido a que por defecto viene en el modelo RGB, en nuestro caso ese obtiene el primer fotograma del video y luego se hace la conversión con la siguiente instrucción.

```
cvtColor(image, image_out, CV_BGR2Lab);
```

4.2. FUNCIONAMIENTO DE TRACKER CON OPENCV

Con la explicación anterior de los modelos de colores, los cuales son usados en el desarrollo del software de reconocimiento de los diferentes elementos indicadores de puntos como masa puntual a analizar, barra de calibración y eje de coordenadas, se puede ahora explicar cómo son usados estos elementos en el software.

4.2.1. Caracterizacion de los indicadores Para empezar debemos dejar en claro y caracterizar de forma adecuada los diferentes indicadores, para esto se tomó la decisión de asignarle un color a cada elemento. Por lo que se asignó el color rojo para la barra de calibración, el color amarillo para el eje de coordenadas y el color azul para la masa que se desea analizar.

De nuevo se debe tener en cuenta que los tonos de los diferentes colores deben estar en contraste para que no se confunda con otros colores, por ejemplo que el azul, no se encuentre rodeado de un color como el verde, ya que puede ocasionar que el software se confunda y no lo identifique por reconocerlo como un verde.

A continuación se van a mostrar en el modelo de color RGB los colores que se van a usar en el software con un determinado rango de aceptación.

Azul (masa puntual):

	R	G	B
	0	56	140
	179	217	255

Rojo (barra de calibración):

	140	0	0
	255	204	230

Amarillo (eje de coordenadas):

	140	0	0
	255	255	191

Estos son los colores que se reconocerán en el video, pero no se hará un continuo reconocimiento del color cuadro a cuadro del video por OpenCV, este reconocimiento solo se hará una vez y luego lo que se hace es un seguimiento de los objetos identificados, esta única vez que se hace el reconocimiento es en el primer cuadro o frame del video, esta identificación se hace con OpenCv, por esto lo primero es importar el video, leer el primer cuadro, guardar esta imagen y luego hacer una identificación por medio del modelo RGB de los valores que se encuentran en esta imagen, estos valores son guardados en una matriz, luego se hace la transformación al modelo de color LAB, y estos valores son guardados en otra matriz diferente.

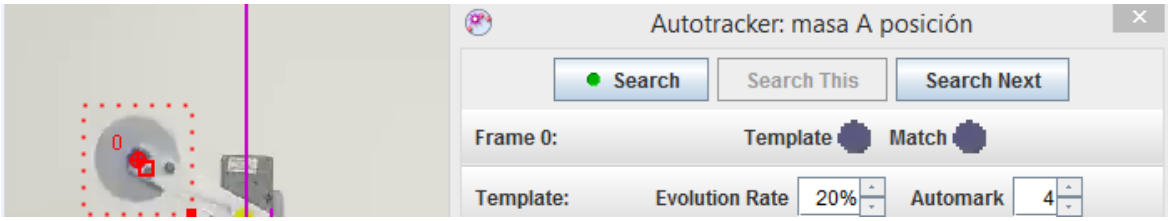
4.2.2. Clasificacion de los colores en el primer frame En este momento donde ya se tienen reconocidos los colores en cada pixel del primer cuadro de video se prosigue a hacer una clasificación del color que se necesita, esto se hace por medio de un elemento llamado condicional, se comparó cada valor que se tiene en el cuadro con el que se desea encontrar, es decir, con el indicador que se desea trabajar, por ejemplo si se está haciendo el reconocimiento del eje coordenado el cual está caracterizado con el color amarillo tomamos el valor en LAB de la imagen y luego se compara con el color en B, que es donde se encuentra ubicado el color amarillo en este modelo de color, y si concuerda con esto, se toma un valor de 1 y si no concuerda se le asigna un valor de 0, por lo que queda una nueva matriz de valores binarios.

4.2.3. Generación de mascara con el color deseado Si colocáramos una imagen que me identificara los valores de esta matriz mostraría una imagen de blancos y negros, donde los blancos son los valores de los 1 y el negro de los valores 0, a esto se le denomina una máscara, ya que se formaría la imagen que contenga el color que estamos buscando en blanco y el resto de elementos que no contengan ese color será negro.

4.2.4. Reducción de ruido en la mascara Al momento de obtener esta matriz de 1 y 0 se va a observar que se encuentran 1 en partes donde se supone que no se deberían presentar estos valores, por esto es necesario hacer un filtro que reduzca el ruido de tal forma que se borren estos valores y solo se tengan en cuenta los importantes, es decir los que estén concentrados en un punto en específico, para esto lo que se hace es borrar donde los valores de los 1 tengan un bajo tamaño en los pixeles. De esta forma se tendrá una imagen donde se encuentra el color que necesitamos, tomamos la región y le generamos sus respectivas coordenadas, para finalmente asignarle un nombre que es con el que luego será enviado al TRACKER el cual le hará el seguimiento y el estudio cinemático respectivo.

4.2.5. Deteccion de la masa puntual Este proceso se hace para los 3 colores necesarios, solo que con unas pequeñas diferencias para cada uno, ya que en el caso del eje de coordenadas solo se necesita seguir los pasos anteriormente mencionados, pero para la masa puntual no solo basta encontrar los valores de X y Y para definir su posición, ya que aquí necesitamos definir un área de búsqueda, por lo que por defecto se da un rango que el software automáticamente generará, con la posibilidad de ser modificada por el usuario, con el cuadro que se ubica en la masa puntual generada.

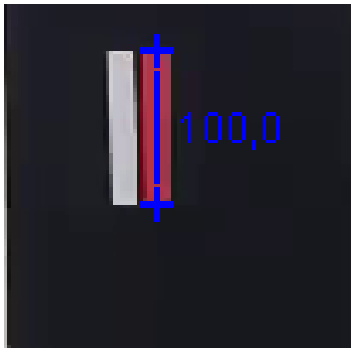
Figura 15. Detección de la masa puntual



Se aconseja que en la masa puntual no se sobrepase el área que contenga el color, debido a que si existe la interferencia con otro objeto, luego en la búsqueda de coincidencia en los siguientes fotogramas, el software podría perderse, este por el lado de la masa puntual.

4.2.6. Deteccion de la barra de calibracion Ahora por la barra de calibración hay que denotar que está compuesta por dos puntos ya que la barra de calibración como tal es una recta, que representa una escala, que le ayuda al software al momento de realizar los cálculos, como está compuesta de dos puntos, es necesario hacer unas modificaciones en el momento de elaborar el código de reconocimiento, al igual que la masa puntual el proceso de reconocimiento de color, pero a diferencia que el anterior indicador, este no solo identificara el punto central de los pixeles reconocido por el color que deseamos, aquí tenemos que identificar dos puntos, en este caso serán el máximo y el mínimo de los pixeles que contengan el color que en este caso es el rojo.

Figura 16. Detección de la barra de calibración.

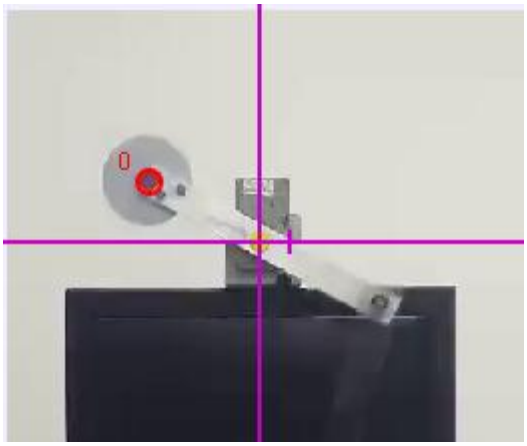


Cuando se vaya a realizo la detección de identificadores, fue importante en el momento de grabar el video colocar la barra de calibración de color rojo de forma vertical, ya que el código se dispuso de esta forma, si en lugar de hacer la grabación con la barra de calibración de forma vertical se hace de forma horizontal habría un error en el funcionamiento del software, ya que reconocería de forma adecuada el color, pero al momento de definir los dos puntos necesarios para generar la barra de calibración, buscaría los extremos de los pixeles que contengan este color por arriba y por abajo no a extremos de derecha ni izquierda, por lo que el software tendría una percepción errada de lo que en realidad es la barra de calibración, y al momento de hacer cálculos, el software recibiría información errónea de la escala, por lo que generaría datos incorrectos.

4.2.7. Deteccion del eje de coordenadas Como último indicador se tiene el eje de coordenadas, el cual nos permite indicarle al software donde tiene que empezar a hacer las comparaciones de posición, se recomienda que este punto de eje sea un punto fijo ya que TRACKER solo tiene disponible el seguimiento de puntos para la masa puntual que es el punto que se desea analizar.

El eje de coordenadas será la marca circular de color amarillo, el software hará la detección de la región donde se encuentre este color y generara el punto en el centro de esta región.

Figura 17. Detección del eje de coordenadas.



Este es el proceso de detección que se programó para hacer el reconocimiento automático de los indicadores necesarios para el correcto funcionamiento del software. Sin embargo, no es la única manera de realizar la detección, otra forma que se consideró fue el reconocimiento de marcas por medio de formas.

4.3. DETECCION DE INDICADORES POR MEDIO DE FORMAS

Una de las posibilidades que se estudió a la hora de resolver el problema de cómo hacer el reconocimiento automático de los identificadores necesarios, fue hacer este reconocimiento por formas, es decir seleccionar una determinada forma geométrica que sea fácil de identificar que representa los elementos necesarios, son requeridas tres figuras geométricas que representen la masa puntual, el eje de coordenadas y la barra de calibración. Las figuras geométricas deben ser sencillas ya que en el momento en el que el usuario del software se dirija a crear el video no debe complicarse al hacerlo con complejas figuras geométricas y también para que se disminuya la posibilidad de que se den errores en el reconocimiento.

Este trabajo es posible hacerlo con la librería OpenCV, ya que se necesita hacer un trabajo con la imagen y esta librería se especializa en hacer el estudio

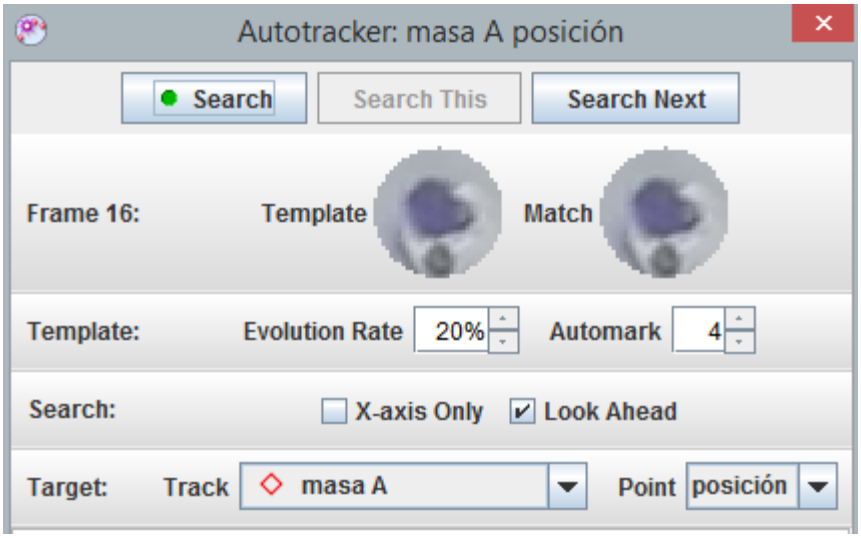
completo de los efectos visuales que pueden ser registrados por una cámara. Aquí el proceso es tomar el primer fotograma del video y guardar esta imagen para luego ser leída en el software y trabajar solo en base a ella, identificando las características necesarias para que el software TRACKER arranque y haga el estudio cinemático del video, en este proceso no se modificara el código de TRACKER ya que este es el encargado de hacer el estudio cinemático y no el reconocimiento de los objetos.

Cabe resaltar que no es necesario los reconocimientos de decenas de figuras geométricas ya que esto generaría una cantidad de líneas innecesarias en el código, solo se requieren tres figuras geométricas que nos identifiquen lo que necesita el TRACKER para su correcto funcionamiento, por esto se planteó como la posibilidad de definir las siguientes figuras:

El círculo funcionaria como la masa puntual, teniendo en cuenta que el TRACKER usa un círculo como área de búsqueda de la masa puntual en todos los fotogramas.

El triángulo se usaría como eje de coordenadas, este fue seleccionado por su sencillez al momento de realizar el video y no complicar de sobremanera al usuario que va a usar el software, es de tener en cuenta que el eje de coordenadas no es un área sino un punto por lo que este punto se generara en el centro del triángulo lo que posiblemente tenga como resultado una pérdida de precisión en los valores deseados pero que pueden ser pasados por alto ya que el error generado puede ser insignificante.

Figura 18. Cuadro de dialogo en TRACKER. AutoTRACKER.



El rectángulo se identificaría como la barra de calibración, aprovechando que este elemento lo que quiere representar es una longitud por lo que facilitaría la guía de mostrar con los extremos de este rectángulo la magnitud de longitud de referencia que se le va a dar al software para que este produzca los cálculos de forma adecuada, esto puede ser observado en la ilustración 12.

Para hablar del reconocimiento de figuras geométricas primero se debe hablar de algunas definiciones importantes que se deben tener claras para lograr un adecuado procedimiento a la hora de realizar el código, uno de estos es el recorrido de una imagen, esto es la forma en que se le hace estudio a una imagen y es de izquierda a derecha y de arriba abajo, pixel por pixel, es de tener en cuenta que se realiza un estudio de color para poder identificar los elementos que la componen y se hace por modelo de color RGB.

4.4. DECISION DE LA FORMA OPTIMA PARA HACER EL RECONOCIMIENTO VISUAL

Luego de haber analizado las dos alternativas de realizar la actualización del software TRACKER, y haber emitido un juicio comparativo de las dos posibilidades, se sacaron las ventajas y desventajas que tiene cada alternativa, para hacer una correcta decisión que permita implementar en el software el código optimo que haga a TRACKER un software más automatizado y realizar una mejora en su desempeño.

4.4.1. Ventajas y desventajas en el reconocimiento de formas Debido a que para el reconocimiento de formas es necesario hacer un reconocimiento de contorno que consiste en detectar todos los bordes que se presenten en el cuadro o frame que se está estudiando y este a su vez se hace detectando el color, se pudo observar que este método tiene una gran desventaja, debido a que para poder detectar la forma primero se debe analizar el color, lo que genera un gran aumento en las líneas de código y complejidad del mismo. Una ventaja es que aumenta la posibilidad de efectuar simultáneamente un análisis de diferentes masas puntuales inclusive del mismo color.

4.4.2. Ventajas y desventajas en el reconocimiento de colores Luego de observar que para realizar el reconocimiento de formas se debe hacer primero la detección colores, esto viene a ser una gran ventaja ya que no es necesario hacer dos tipos de detección, lo cual hace que se simplifique un poco más el problema, además que las líneas de código se reducen, una desventaja es la dificultad que se presenta al momento de hacer un análisis simultaneo de varias masas puntuales, ya que se debe tener cuidado de la cantidad de colores que se deben analizar en un solo cuadro, puesto que el ingresar más cantidades de colores hace que el software se sature y pueda confundir los diferentes colores que conforma cada pixel.

4.4.3. Alternativa implementada Luego de sopesar las ventajas y desventajas de cada una de las alternativas se tomó la decisión de que el mejor método para realizar la inclusión del sistema de detección de identificadores para la automatización del software de análisis cinemático de video TRACKER, es la detección de reconocimiento por medio de la detección del color, ya que presenta una disminución en la complejidad al momento de abordar el problema, además de la reducción del código hecho en eclipse.

Teniendo en cuenta la desventaja que presenta con respecto a la otra opción, que es la de no tener que hacer un análisis simultaneo de diferentes masas puntuales, pero si es posible realizar en un mismo archivo; el análisis cinemático de un mismo video, solo se debe realizar para la primera masa puntual, y luego volverla a hacer para la segunda, esto se repetirá según la cantidad de masas puntuales que se tengan.

De esta forma se logra poder tener un método con el cual se calculan los parámetros necesarios para enviar al módulo TRACKER del software desarrollado.

5. MODULO TRACKER

En este capítulo se toca a fondo el funcionamiento de TRACKER y su manera de operar, también se habla de los algoritmos de cálculo con los que trabaja el software al momento de realizar el análisis cinemático de los videos, para así lograr comprender la razón de ser de los resultados que arroja el software al momento de realizar el análisis.

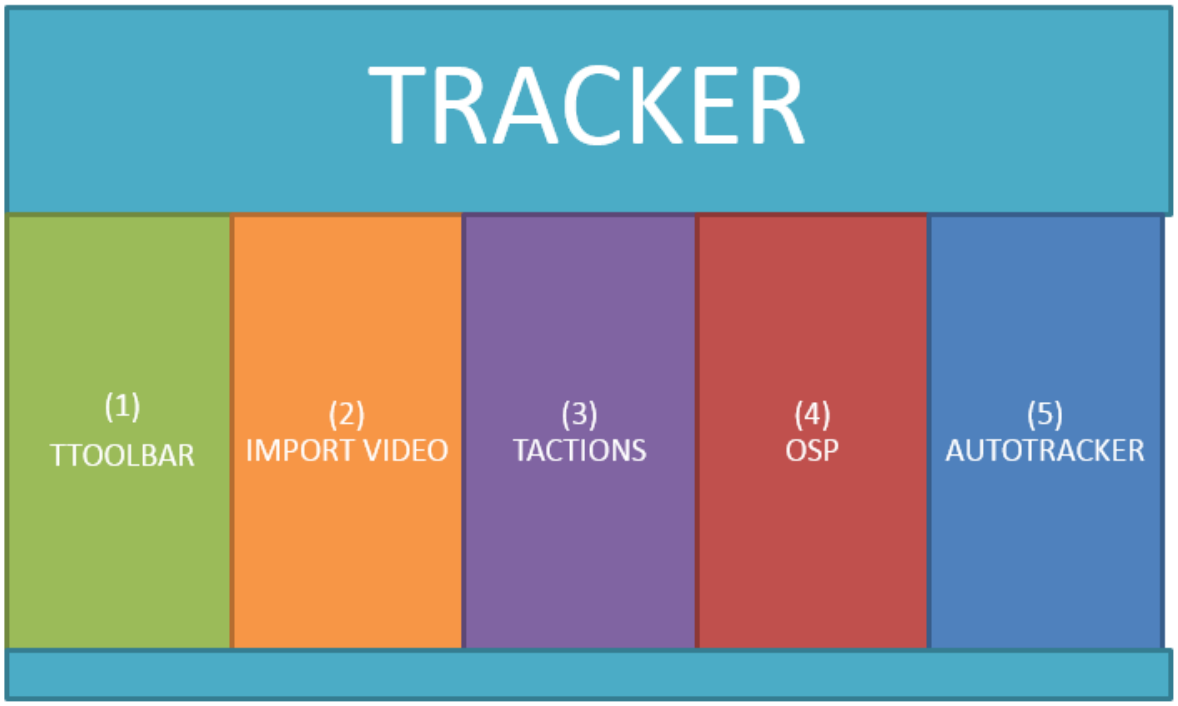
El código de TRACKER se programó en lenguaje JAVA, este lenguaje es una forma de programar orientada a objetos, que nos permite clasificar nuestro código en clases, cada clase cumple una función específica en cada proyecto, por ejemplo TRACKER presenta clases que tienen relacionados los diferentes componentes que posee en su interfaz, otras clases contienen el código extraído de Open Source Physcs, donde se encuentra la parte de los algoritmos matemáticos que tiene en su interior el software.

El módulo *TRACKER* presenta una amplia lista de clases que componen su software, y estas clases se agrupan para dar orden y formar así unas clases generales de las cuales se compone el software y la cual cubre un frente de interés a la hora del desarrollo de código.

Se modificaron específicamente 3 clases, la clase que controla la interfaz es llamada TTOLBAR, es la que permite tener una visualización del software TRACKER, en el momento en que se está trabajando, es por esto que se necesita un código especifico que controle estas funciones, otra clase modificada es la de TACTIONS, la cual permite crear un enlace de las acciones que tomamos a la hora de interactuar con la interfaz y por último AUTOTRACKER la cual es la clase encargada de realizar el seguimiento de los puntos de interés durante el análisis

de video, estas son las clases que en el capítulo 6 se modificaron para dar paso al nuevo producto.

Figura 19. Diagrama general de TRACKER.



Otras clases también muy importantes que posee TRACKER son OSP (Open Source Physics) la cual posee las librerías de los algoritmos matemáticos con el cual es posible hacer el análisis de datos que arroja TRACKER, es decir, con los datos de posición, se realiza un tratamiento a esos datos para determinar características como la velocidad y aceleraciones lineales, otra clase es IMPORT VIDEO la cual otorga la posibilidad al software de trabajar con un motor de video, para este caso los dos motores de video posibles son QuickTime y Xuggle, los cuales son fundamentales, ya que este software trabaja con videos.

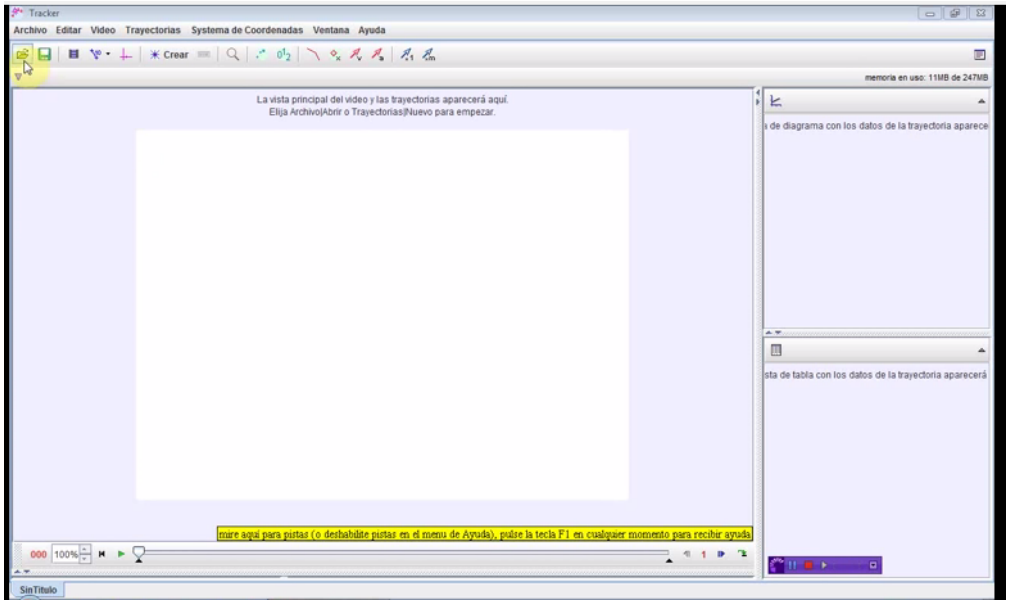
5.1. PASOS DE EJECUCIÓN PARA EL MANEJO DE TRACKER

TRACKER es un software de análisis de videos en el cual se hace un estudio cinemático, este software está programado con la ayuda de la herramienta Open Source Physics, la cual sirve como apoyo para el desarrollo practico del estudio cinemático de un movimiento en 2 dimensiones, su funcionamiento es el siguiente.

Pasos a seguir para un correcto desarrollo del software:

PASO 1: Conocer la interfaz del software, en la cual se tendrá que hacer clic en abrir y luego importar el video que se desea analizar.

Figura 20. Interfaz de TRACKER.

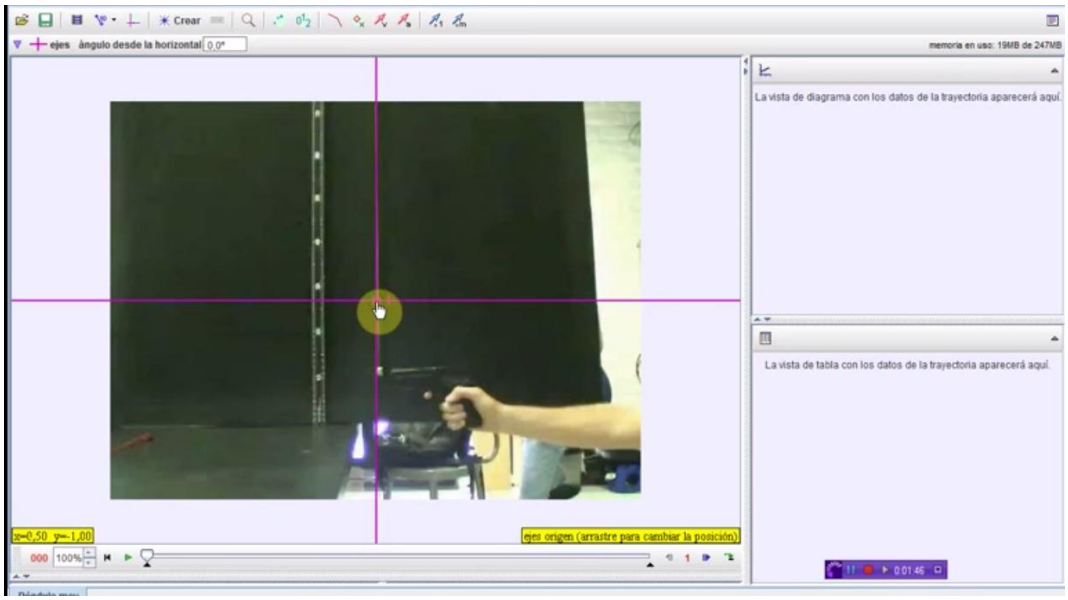


NOTA: Luego de haber importado el video aparecerá el primer fotograma del mismo, en el video es conveniente que el objeto a analizar tenga buen contraste con respecto al fondo para que el software detecte fácilmente el punto de interés y no se pierda.

Posteriormente se pasa a definir los puntos de referencia necesarios para el correcto funcionamiento del software.

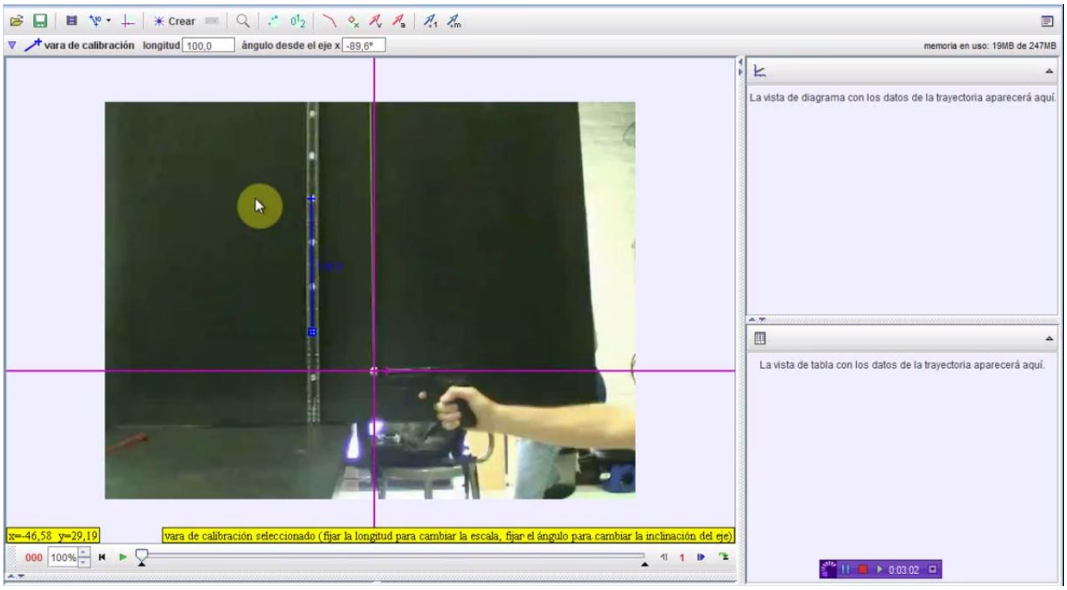
PASO 2: Establecer el eje coordenado, con el cual a partir de él se realiza el respectivo análisis, esto se hace dando clic en el botón ejes como lo indica la figura y luego colocándolo sobre el punto fijo donde se desea que sea el eje de coordenadas.

Figura 21. Ejemplo del establecimiento del eje de coordenadas.



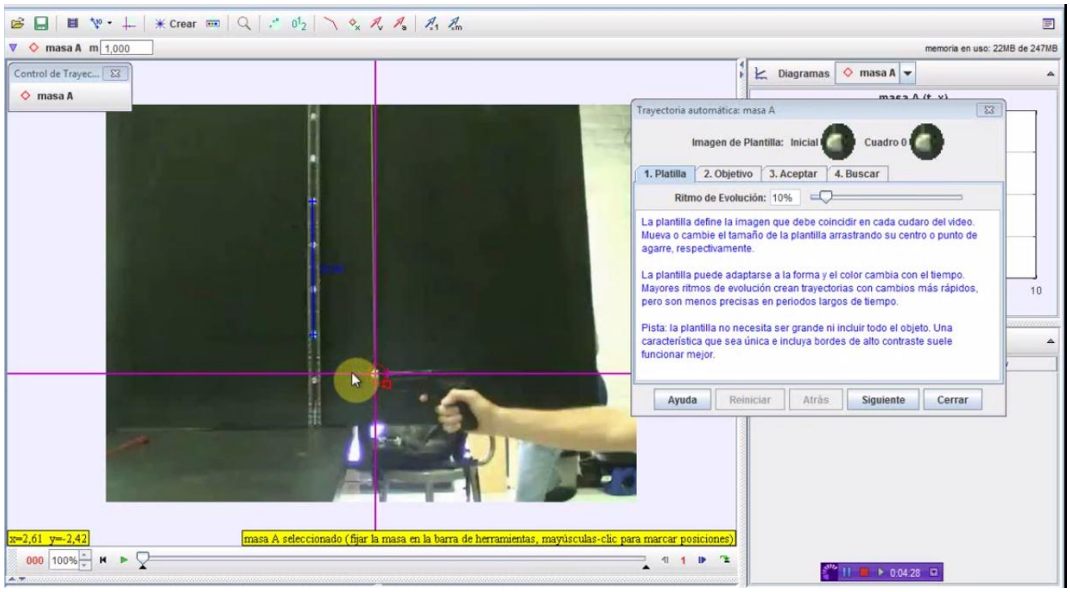
PASO 2: Luego se necesita que exista en el video una distancia conocida, es decir una distancia real que sirva de referencia de cuanto es el desplazamiento que está recorriendo el objeto a analizar en una unidad conocida, y la cual hace que sirva como una distancia de calibración que le ayuda al software a interpretar su movimiento de forma más precisa. Esto se hace dando clic en la herramienta calibración cinta métrica.

Figura 22. Ejemplo del establecimiento de la barra de calibración.



PASO 3: Se va a definir el objeto o masa a la cual se le desea desarrollar el análisis cinemático y a su vez se le hará una sugerencia al software de donde debe buscar la masa en los siguientes cuadros, para esto se le da clic en crear y luego clic en masa puntual y con este nuevo cursor daremos ahora clic en el objeto a analizar, luego se dará clic en trayectoria automática.

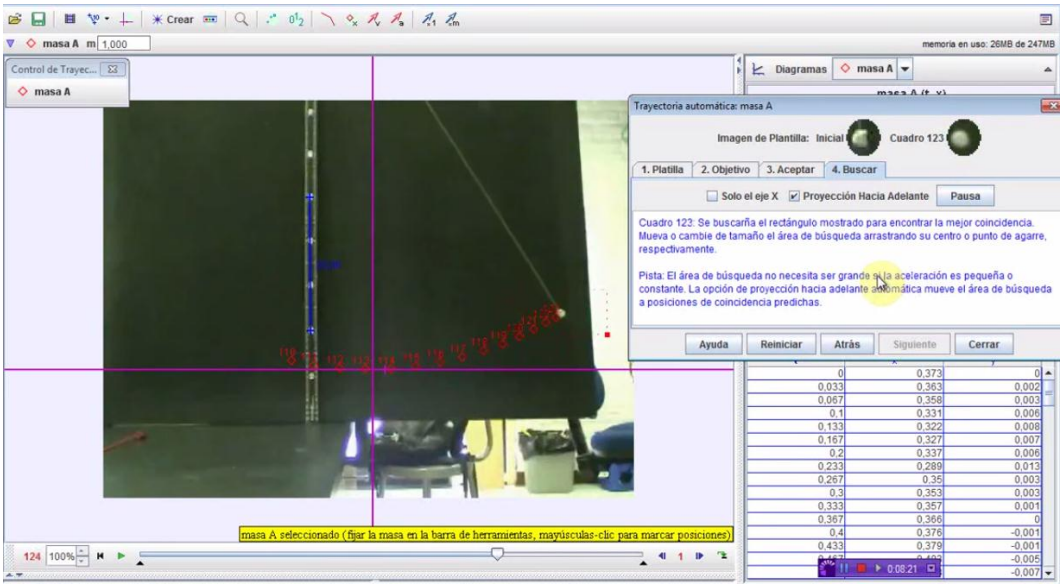
Figura 23. Ejemplo del establecimiento de la masa puntual.



PASO 4: Luego se le da clic a buscar y el software buscará cuadro a cuadro el objeto que se seleccionó para analizar con una aceptación en el rango que se le haya puesto. Si por algún caso en algún cuadro no se encuentre el objeto a analizar se puede hacer de forma manual o también si se encuentra el objeto pero con una aceptación por debajo a la que se le indico el software le preguntara al usuario si este está correcto o no.

Luego de analizar el video cuadro a cuadro los resultados se verán en una tabla, la cual relaciona posición de cada uno de los ejes en cada tiempo, con su respectiva grafica correspondiente, y además de eso se puede solicitar la gráfica con respecto a lo que se desee encontrar ya sea posición, velocidad angular o aceleración angular de la masa puntual.

Figura 24. Ejemplo del análisis cinemático.



5.2. ALGORITMO DE DIFERENCIAS FINITAS.

El método de diferencias finitas es un método que otorga la posibilidad de tener una aproximación de diferentes ecuaciones diferenciales en determinadas derivadas parciales que se desenvuelven en espacios conocidos. Este procedimiento se efectúa con el motivo de brindarle a las ecuaciones una determinada sencillez para lograr resolverlas, es de aclarar que este procedimiento es adecuado, permitido y da como respuesta datos acertados.

El procedimiento para la aplicación de este método consiste en discretizar el espacio en el que se desea resolver la ecuación por medio de una malla que se encuentra separada una distancia fija en ambas direcciones.

Es posible desarrollar una serie de Taylor alrededor de ese punto lo que genera una expansión, partiendo de una función definida en (a, b), la cual presenta hasta

la k-esima derivada, presenta la expansión f(x) usando series de Taylor alrededor de un punto cualquiera x_i, el cual se encuentra contenido en el intervalo.

Hay diferentes formas de realizar una aproximación a la primera derivada, las cuales puede entregar una mayor o menor precisión, esto depende del esfuerzo computacional, está el método de diferencias progresivas, el de diferencias regresivas y diferencias centradas. Se utilizó esta última puesto que si se obtuviera una mayor cantidad de puntos en el espacio en el que se está trabajando se logra generar la construcción de fórmulas más precisas. Esta es la situación que se encuentra con la velocidad, es por esto que el método para encontrar la velocidad para TRACKER es el siguiente:

$$v_i = \frac{x_{i+1} - x_{i-1}}{2 * dt} \quad (1)$$

De igual forma se necesita trabajar con derivadas de segundo orden por lo que se procede un desarrollo a partir de la serie de Taylor; sumando las ecuaciones anteriores y realizándole una depuración de las primeras derivas se encuentra con la expresión que se desea y es con la que es posible encontrar la aceleración en este software.

$$a_i = \frac{2 * x_{i+2} - x_{i+1} - 2 * x_i - x_{i-1} + 2 * x_{i-2}}{7 * dt^2} \quad (2)$$

Es importante aclarar que las expresiones anteriormente mostradas son aproximaciones que se realizan para lograr dar una respuesta al problema físico planteado, es por esto que al poner en marcha el trabajo del software los resultados no serán totalmente exactos, pero si aproximados ya que se siguió un método de análisis de datos de forma correcta.

5.2.1 Ejemplo Se mostrará la forma para calcular la velocidad y la aceleración lineal instantánea del punto que conecta la manivela con la biela; para hallar estos datos en un instante de tiempo se necesitan dos datos de posición, la posición en el instante anterior y la posición del instante siguiente; puesto que no tenemos estos datos para el tiempo cero se calculará la velocidad a partir del tiempo 0,033.

Obtenemos el primer valor de velocidad de la siguiente manera:

Tabla 1. Ejemplo de diferencias finitas, primeros 3 datos de posición.

	t [seg]	x [cm]	y [cm]
i-1	0,000	16,000	-8,81
i	0,033	15,600	-9,39
i+1	0,067	15,300	-9,97

En la tabla 1 tenemos los datos necesarios para hallar la velocidad lineal en el instante "i" (no se muestran todos los decimales en la tabla), los valores de posición en el eje "x" y el eje "y" son determinados por las posiciones marcadas por TRACKER, ahora reemplazando en la ecuación (1) hallamos $v_{x\,i}$ y $v_{y\,i}$

$$v_{x\,i} = \frac{15{,}3 - 16}{2 * 0{,}033} = -10{,}60$$

$$v_{y\,i} = \frac{-9{,}97 - (-8{,}31)}{2 * 0{,}033} = -17{,}57$$

Ahora hallamos la magnitud de la velocidad

$$v_i = \sqrt{{v_{x\,i}}^2 + {v_{y\,i}}^2} = \sqrt{-10{,}60^2 + -17{,}57^2}$$

$$\boldsymbol{v_i = 20{,}52\ [cm/s]}$$

Para el cálculo de la aceleración utilizamos 5 puntos que corresponderán a los datos de posición de dos instantes anteriores de tiempo, el instante de tiempo en el cual queremos hallar la aceleración y los dos instantes siguientes.

Los datos necesarios para determinar la aceleración en el instante "i" se muestran en la tabla 2.

Tabla 2. Ejemplo de diferencias finitas, primeros 5 datos de posición.

	t [seg]	x [cm]	y [cm]
i-2	0,000	16,00	-8,81
i-1	0,033	15,60	-9,39
i	0,067	15,30	-9,97
i+1	0,100	14,80	-10,60
i+2	0,133	14,4	-11,20

Reemplazando en la ecuación (2) hallamos $a_{x\,i}$ y $a_{y\,i}$

$$a_{x\,i} = \frac{2 * 14{,}4 - 14{,}8 - 2 * 15{,}3 - 15{,}6 + 2 * 16}{7 * 0{,}033^2} = -26{,}23$$

$$a_{y\,i} = \frac{2 * (-11{,}2) - (-10{,}6) - 2 * (-9{,}97) - (-9{,}39) + 2 * (-8{,}81)}{7 * 0{,}033^2} = -11{,}80$$

Ahora hallamos la magnitud de la velocidad

$$a_i = \sqrt{{a_{x\,i}}^2 + {a_{y\,i}}^2} = \sqrt{-26{,}23^2 + -11{,}80^2}$$

$$\boldsymbol{a = 28{,}76\ [{cm}/{s^2}]}$$

Aplicando las mismas ecuaciones se obtienen las velocidades y aceleraciones para todos los instantes de tiempo.

6. AUTOMATIZACIÓN DEL SOFTWARE TRACKER.

En el capítulo 4 se habló sobre cómo realizar una adecuada detección de los diferentes indicadores o marcas que necesita el software TRACKER para desarrollar su operación de análisis cinemáticos de videos, luego de haber decidido la forma más conveniente de realizar esa detección, la cual fue por medio de la detección de colores con la ayuda de la librería OpenCV, la cual al dar un clic se dibuja en el primer cuadro de los diferentes fotogramas las marcas de masa puntual, barra de calibración y eje de coordenadas.

6.1. AUTODETECTED

Ahora se procederá a hacer una explicación de la clase agregada al software, la cual permitió realizar una mejora a este software añadiéndole elementos de detección automática, aplicándole colores a diferentes puntos en el video. Esta clase se llama AUTODETECTED y fue programada en lenguaje JAVA por medio de la plataforma de desarrollo de entornos ECLIPSE.

Como primera medida se hace un procesamiento de imagen donde se hace una conversión de la imagen de bytes del primer cuadro a uno de valores enteros, creando luego una matriz del mismo tamaño de la primera donde quedaran guardados esos valores, para luego realizar la conversión de los valores encontrados en RGB y pasarlos a valores del modelo de color LAB y también se llama la librería OpenCV, para poder trabajar con las herramientas de análisis de imagen.

```
/*public static void imageView(BufferedImage imagen) {
                System.loadLibrary("opencv_java2410");
```

```
                displayImage(imagen);
        }*/
        //convierte la imagen del video de de bytes a enteros
        public static Mat matify(BufferedImage im) {
            // Convert INT to BYTE
                BufferedImage convIm = new BufferedImage(im.getWidth(),
im.getHeight(),BufferedImage.TYPE_3BYTE_BGR);
                //imageProcessing(im);
                convIm.getGraphics().drawImage(im, 0, 0, null);
            // Convert bufferedimage to byte array
            byte[] pixels = ((DataBufferByte)
convIm.getRaster().getDataBuffer()).getData();

            // Create a Matrix the same size of image
            Mat image = new Mat(convIm.getHeight(), convIm.getWidth(),
CvType.CV_8UC3);
            // Fill Matrix with image values
            image.put(0, 0, pixels);

            return image;

        }
        //Funcion que procesa la imagen
        public static List<Mat> imageProcessing(BufferedImage imagen) {
                System.loadLibrary("opencv_java2410");
                //imageProcessing(imagen);
                Mat bgrImage = matify(imagen);
                Mat labImage = matify(imagen);
                Imgproc.cvtColor(bgrImage, labImage, Imgproc.COLOR_BGR2Lab);
```

```
        List<Mat> lab_channels = new ArrayList<Mat>();
        Core.split(labImage, lab_channels);
        return lab_channels; }}
```

La detección de los contornos se crean generando una matriz para guardar la información de la primera imagen que tiene el video, por esta razón también es necesario llamar la primera imagen imgproc, luego se hace una búsqueda de los contornos en esta imagen lo que genera las matrices con datos de dichos contornos donde es guardada.

```
public static Mat rmSmallObjects(Mat im, double size){

        // Only accept CV_8UC1
    if (im.channels() != 1 || im.type() != CvType.CV_8UC1)
        return im;

    // Find all contours
    Mat imCopy=im.clone();
    List<MatOfPoint> contours = new ArrayList<MatOfPoint>();
    Imgproc.findContours(imCopy,contours, new Mat(), Imgproc.RETR_LIST,
Imgproc.CHAIN_APPROX_SIMPLE);

    for (int i = 0; i < contours.size(); i++)
    {
        // Calculate contour area
        double area =Imgproc.contourArea(contours.get(i));

        // Remove small objects by drawing the contour with black color
        if (area > 0 && area <= size)
            Imgproc.drawContours(im, contours, i, new Scalar(0,0,0),-1);
    }
```

```
        return im;
    }
```

Se realiza también un barrido de información, si la zona donde se generó un contorno tienen el área muy pequeña al tamaño predefinido se debe pasar por alto este contorno, es por esto que se clona la matriz donde quedaran guardados los contornos con las pequeñas áreas borradas, las cuales se habían detectado anteriormente como contornos.

El primer marcador en el software es llamar al elemento sobre el cual se va a trabajar, este es el primer cuadro en el histograma de los fotogramas que se encuentran presentes en el video, luego se necesita guardar la información que en ella se encuentran de los diferentes valores RGB, esto se hace en una matriz por lo cual es necesaria su creación, luego se debe igualmente leer la imagen del primer cuadro en LAB que es el modelo de color con el que finalmente se trabajó.
Se genera una conversión de valores en RGB a LAB y luego se divide la matriz LAB en tres, cada una contiene los valores de sus características por separado L, A y B.

```
//Funcion que procesa la imagen original
   public static List<Mat> imageProcessing(String img) {
       System.loadLibrary("opencv_java2410");
            Mat bgrImage = Highgui.imread(img);//Cargar la imagen bgr
            Mat labImage = Highgui.imread(img);//Cargar la imagen que se convertirá en LAB
            Imgproc.cvtColor(bgrImage, labImage, Imgproc.COLOR_BGR2Lab);//Funcion para convertir imagen de BGR a LAB

            List<Mat> lab_channels = new ArrayList<Mat>();//lab chanels contiene las matrices L, A y B por separado
```

```
            Core.split(labImage, lab_channels);//la funcion split divide la matriz en tres matrices
            return lab_channels;
        }
```

6.1.1. Circulo amarillo Luego se procede a realizar la detección de los diferentes indicadores, el primero con el cual se trabajo es la bola amarilla, el cual en el software es tomado como el eje de coordenadas.

Se identificó cuáles eran los valores en el modelo de color LAB del color amarillo, así mismo en el modelo RGB, luego se genera un clon de la imagen de trabajo para evitar redefinirla más adelante en el código, se procedió a realizar una segmentación de la imagen donde se identifican los pixeles donde se encuentra el color amarillo y se le asigna el valor de 1, si no se encuentra un rango de color se le asignará un valor de 0 a ese pixel, Es posible que en diferentes pixeles se detecte color aunque no lo halla por lo que es necesario eliminar ese ruido generado, por esto se vio la necesidad de eliminar los valores de donde la cantidad de pixeles fuera menos que 100. La imagen de contorno que contiene el color amarillo con la reducción de ruido es guardada, luego se identificó la mayor área generada de color amarillo, se aproximó a un polinomio y se posiciono en el punto central el punto que lo identifica como el eje de coordenadas.

```
// Deteccion bola amarilla
public static Point coordinatesYellowBall(BufferedImage imagen) {
    //L=0
    //A=1
    //B=2
    try{

            List<MatOfPoint> contours = new ArrayList<MatOfPoint>();
```

```
List<Mat> lab_channels = imageProcessing(imagen);
Mat BolaAmarilla=lab_channels.get(2).clone();//se clona la banda B para evitar redefinirla de nuevo
Imgproc.threshold(lab_channels.get(2), BolaAmarilla, 155, 255, Imgproc.THRESH_BINARY);//convierte a 0 y 1 en bola amarilla dependiendo del los pixeles que esten en el rango amarillo
rmSmallObjects(BolaAmarilla, 100);//remueve objetos con tamaño menor a 100 pixeles
Imgproc.findContours( BolaAmarilla, contours, new Mat(), Imgproc.RETR_TREE,Imgproc.CHAIN_APPROX_SIMPLE, new Point(0, 0) );//Guarda los contornos de bola amarilla en contours
    MatOfPoint2f contourBAM_poly = new MatOfPoint2f();
    int indxContour=0;//Guarda el indice del contorno con mayor area
    double maxArea=0.0;
    for(int i=0; i<contours.size();i++){//el for recorre la matriz contours
        double area =Imgproc.contourArea(contours.get(i)); //Devuelve el area
        if (area>maxArea){
            indxContour=i;
            maxArea=area;
        }
    }
    Imgproc.approxPolyDP(new MatOfPoint2f(contours.get(indxContour).toArray()), contourBAM_poly, 3, true);//Se aproxima el contorno a un polinomio (valores continuos)
    Point centerBAM=new Point();
    float[] radiusBAM=new float[1];
```

```
            Imgproc.minEnclosingCircle(contourBAM_poly,          centerBAM,
radiusBAM);//Busca el circulo mas pequeño que encierra el polinomio(contorno)
        return centerBAM;
        }catch(Exception e){
                return null;
        }

        }
```

6.1.2. Circulo azul El siguiente punto de análisis es el del marcador azul que en este caso representa la masa puntual, este se podría decir que es el elemento más importante y con el que se debe tener más cuidado ya que es el elemento de búsqueda de TRACKER, en donde su correcta posición hace que los valores generados luego de su análisis cinemático sean más cercanos a lo deseado ya que si se produce una perdida durante el análisis de video hará que el error que se genere sea alto.

De nuevo se hizo el procesamiento de la imagen, la cual es guardada en el modelo LAB; y se obtienen los valores de este color que nos represente la gama de azules y damos valores de 1 a los que los contengan y 0 a los que no, para la creación del color azul se deben crear 3 objetos, uno de estos es la región de búsqueda por lo que se predefine el área donde se le indicará al software donde debe buscar en el siguiente fotograma al momento de empezar el análisis cinemático del software TRACKER.

```
//BOLA AZUL COORDENADAS
public static Point coordinatesBlueBall(BufferedImage imagen) {

        try{
                List<Mat> lab_channels = imageProcessing(imagen);
```

```
Mat  BolaAzul=lab_channels.get(2).clone();
Core.inRange(lab_channels.get(2), new Scalar(91), new Scalar(115),
BolaAzul);
    rmSmallObjects(BolaAzul, 100);
    int morph_size=3;
int morph_size2=3;
Mat element=Imgproc.getStructuringElement(1, new
Size(2*morph_size + 1, 2*morph_size+1 ));
Mat element2=Imgproc.getStructuringElement(0,new Size(
2*morph_size2 + 1, 2*morph_size2+1 ));
```

El proceso de la detección de contorno es el mismo que el descrito anteriormente en la detección de la bola amarilla, solo que el punto que se genera en el centro del área encontrada no se define como el centro de coordenadas sino como el primer punto donde se encuentra la masa puntual.

```
Imgproc.morphologyEx(BolaAzul, BolaAzul, 1, element );
Imgproc.erode(BolaAzul,BolaAzul,element2);
List<MatOfPoint> contours1 = new ArrayList<MatOfPoint>();
    Imgproc.findContours( BolaAzul, contours1, new Mat(),
Imgproc.RETR_TREE,Imgproc.CHAIN_APPROX_SIMPLE, new Point(0, 0) );
    MatOfPoint2f contourBA_poly = new MatOfPoint2f();
    int indxContour=0;
double maxArea=0.0;
for(int i=0; i<contours1.size();i++){
    double area =Imgproc.contourArea(contours1.get(i));
    if (area>maxArea){
        indxContour=i;
```

```
                    maxArea=area;
                }
            }
                Imgproc.approxPolyDP(new
MatOfPoint2f(contours1.get(indxContour).toArray()), contourBA_poly, 3, true);
                Point centerBA=new Point();
            float[] radiusBA=new float[1];
            Imgproc.minEnclosingCircle(contourBA_poly, centerBA, radiusBA);

            return centerBA;
        }catch(Exception e){
            return null;
        }
    }
```

Además de eso se debe identificar la región de búsqueda del marcador, por lo que es necesario encontrar la región donde están los pixeles que se encuentran con valores de 1 y definirles un tamaño de radio predefinido, el cual es tomado del área máxima la cual fue acercada a la forma de un polinomio, esta área es la mostrada al desplegarse el cuadro de dialogo AUTOTRACKER.

```
    //BOLA AZUL RADIO
    public static double radiusBlueBall(BufferedImage imagen) {

        try{
            List<Mat> lab_channels = imageProcessing(imagen);

                Mat  BolaAzul=lab_channels.get(2).clone();
                Core.inRange(lab_channels.get(2), new Scalar(91), new
Scalar(115), BolaAzul);
```

```
                    rmSmallObjects(BolaAzul, 100);
                    int morph_size=3;
                int morph_size2=3;
                Mat element=Imgproc.getStructuringElement(1, new
Size(2*morph_size + 1, 2*morph_size+1 ));
                Mat element2=Imgproc.getStructuringElement(0,new Size(
2*morph_size2 + 1, 2*morph_size2+1 ));
                    Imgproc.morphologyEx(BolaAzul, BolaAzul, 1, element );
                Imgproc.erode(BolaAzul,BolaAzul,element2);
                List<MatOfPoint> contours1 = new ArrayList<MatOfPoint>();
                    Imgproc.findContours( BolaAzul, contours1, new Mat(),
Imgproc.RETR_TREE,Imgproc.CHAIN_APPROX_SIMPLE, new Point(0, 0) );
                    MatOfPoint2f contourBA_poly = new MatOfPoint2f();
                    int indxContour=0;
                double maxArea=0.0;
                for(int i=0; i<contours1.size();i++){
                    double area =Imgproc.contourArea(contours1.get(i));
                    if (area>maxArea){
                            indxContour=i;
                            maxArea=area;
                    }
                }
                    Imgproc.approxPolyDP(new
MatOfPoint2f(contours1.get(indxContour).toArray()), contourBA_poly, 3, true);
                    Point centerBA=new Point();
                float[] radiusBA=new float[1];
                Imgproc.minEnclosingCircle(contourBA_poly, centerBA, radiusBA);
                    double a = 0.8;
                return (radiusBA[0] * a);
        }catch(Exception e){
```

```
            return -1;
        }
```

6.1.3. Cinta métrica El otro marcador que se genero fue la cinta métrica, la cual fue identificada por una barra de color rojo, nuevamente se procesa la primera imagen de los fotogramas, se hace la conversión de colores de RGB a LAB guardando los valores en una matriz donde fueron detectados los valores que genera la gama de rojos, se hace la detección de contornos donde se encuentre este color, se aproxima a un polinomio, Debido a que se espera un rectángulo se toma el rectángulo encontrado más pequeño y se encierra, en este rectángulo se definen los puntos vitales, como lo son el punto superior y el punto inferior.

```
    // CINTA METRICA
        public static Point[] coordinatesRedTape(BufferedImage imagen) {
            try{
                List<Mat> lab_channels = imageProcessing(imagen);

                Mat  BarraRoja=lab_channels.get(1).clone();
                Imgproc.threshold(lab_channels.get(1), BarraRoja, 155, 255, Imgproc.THRESH_BINARY);
                List<MatOfPoint> contours2 = new ArrayList<MatOfPoint>();
                Imgproc.findContours( BarraRoja, contours2, new Mat(), Imgproc.RETR_TREE,Imgproc.CHAIN_APPROX_SIMPLE, new Point(0, 0) );
                MatOfPoint2f contourBR_poly = new MatOfPoint2f();
                Imgproc.approxPolyDP(new MatOfPoint2f(contours2.get(0).toArray()), contourBR_poly, 3, true);
                Rect boundRectBR = Imgproc.boundingRect( new MatOfPoint(contourBR_poly.toArray()) );//LIMITA AL RECTANGULO MAS PEQUEÑO Y LO ENCIERRA
```

```
                Point ini = new Point();//limite superior
                Point fin = new Point();//limite inferior
                Point centro = new Point();//punto central
                double[] Endsini =
{(boundRectBR.x+boundRectBR.width/2),boundRectBR.y};
                double[] centr =
{(boundRectBR.x+boundRectBR.width/2),(boundRectBR.y+boundRectBR.height/2
)};
                double[] Endsfin =
{(boundRectBR.x+boundRectBR.width/2),(boundRectBR.y+boundRectBR.height)};
                ini.set(Endsini);//se agregan el punto inicial
                fin.set(Endsfin);//Se agrega el punto central
                centro.set(centr);//se agrega el punto final
                Point[] centerBr={ini, centro, fin};
                //{inicio, centro , fin}
                    return centerBr;

            }catch(Exception e){
            return null;
        }
        }
```

6.2. TTOOLBAR

Otro elemente que se tuvo que modificar fue el lugar donde se incluyó la opción de dar clic en las mejoras que se ofrecieron en el software, ya que se dejó la posibilidad de operar con el modo normal de trabajo que viene usando TRACKER, y con las mejoras implementadas, incluso se puede trabajar como un hibrido, ya

que es posible trabajar con el sistema de auto detección de las marcas junto con la posibilidad de modificar o mover los marcadores, esto es posible si se desea.

En este apartado se puede observar cómo se incluyó el botón llamado AUTObutton y como quedo habilitado para su funcionamiento en la barra de tareas del software TRACKER, el icono fue creado como una lupa y fue agregado en las imágenes del software, donde quedo guardado para que cada vez que se cargue el software, este aparezca.

El botón AUTObutton quedo configurado para que al momento de dar clic, dibuje en el primer cuadro las marcas seleccionadas que son los puntos que se necesitan de eje de coordenadas, barra de calibración y masa puntual, si se desea ocultar solo se necesita dar de nuevo clic en este botón (AUTObutton) y automáticamente estas marcas quedaran invisibles, de igual forma si se desea que vuelvan a aparecer solo hay que dar de nuevo clic en el mismo icono.

TTOOLBAR:

```
// AUTO button muza
    AUTOButton = new TButton(AUTOOffIcon,AUTOOnIcon); // muza
    AUTOButton.addActionListener(actions.get("AUTOVisible"));       //$NON-NLS-1$
muza
```

6.3. AUTOTRACKER

La clase AutoTRACKER fue modificada ya que se necesitaba indicarle al software donde iban a quedar seleccionados las marcas que anteriormente habían sido detectadas, es por esto que se indican las coordenadas (x, y) del punto central de la masa puntual, también el radio donde queda indicada la región de búsqueda, este radio se crea a partir de las coordenadas anteriormente mencionadas. Otro

aspecto que se modifica es el área de la región de búsqueda, el tamaño que se dibuja de estos elementos se hace por defecto, el mismo que determina el software, de nuevo se menciona que es posible modificarlo, si el usuario de *Kine-UIS* lo considera conveniente.

AUTOTRACKER

```
 //muza
 protected void addKeyFrame_pointMassAuto(TPoint p, double x, double y, double radius) {
                int n = TRACKERPanel.getFrameNumber();
                Target target = new Target();
                Shape mask = new Ellipse2D.Double();
                maskCenter.setLocation(x, y);
                maskCorner.setLocation(x+radius,  y+radius);//modifica el area del Template
                searchCenter.setLocation(x, y);
                searchCorner.setLocation(x+(radius*2),  y+(radius*2));//modifica el area del Search
                Map<Integer, FrameData> frames = getFrameData();
                KeyFrame keyFrame = new KeyFrame(p, mask, target);
                frames.put(n, keyFrame);
                clearSearchPointsDownstream();
                refreshSearchRect();
            refreshKeyFrame(keyFrame);
            getWizard().setVisible(true);
//          getWizard().refreshGUI();
//          search(false, false); // don't skip this frame and don't keep stepping
                        TRACKERPanel.repaint();
        }
 //muza
```

6.4 TACTIONS

Y por último, la clase que se necesitó modificar es la de TACTIONS donde se hace un enlace de lo nuevo que se creó con lo usado por TRACKER. Aquí solo se necesitó modificar la barra de calibración (Tape measure) y la masa puntual (Point mass) ya que el eje de coordenadas era simplemente un punto el cual no era modificado, en cambio la barra de calibración se configuró para que la aparición de esta se dispusiera siempre de la misma forma vertical; y la masa puntual cabe recordar que está compuesta por 3 elementos los cuales variaron en su composición de aparición, ya que por defecto TRACKER los mostraba en el centro del primer frame, ahora aparecen en el lugar donde fue colocada la marca.

TACTIONS:

```
public class TActions {

    static boolean VCname = true;// muza
    static TapeMeasure tapeMeasureAuto = new TapeMeasure();// muza
    static PointMass pointMassAuto = new PointMass();//muza
```

7. ANALISIS CINEMATICO

Se debe hacer el análisis cinemático del mecanismo para lograr determinar de qué manera se logra encontrar la velocidad angular y la aceleración angular, es por esto que se explicará que método se usó para encontrar estos dos datos que finalmente son el objetivo de este proyecto, por lo que se mostrará y explicara las ecuaciones usadas a lo largo de este libro; todas las unidades mostradas en este capítulo se encuentran en centímetros y segundos.

Específicamente se hablara del estudio cinemático de un mecanismo manivela – biela – corredera, ya que este fue el mecanismo seleccionado sobre el cual probar el software TRACKER para encontrar los datos de velocidad angular y aceleración angular.

La cinemática como rama de la física que estudia el movimiento de los cuerpos en función del tiempo, despreciando las fuerzas como causas que producen dicho movimiento pero si hacen el estudio completo de las consecuencias que estas conllevan por lo que es importante en nuestro mecanismos realizarle el estudio para ver qué características tiene el movimiento de cada uno de los elementos que lo componen.

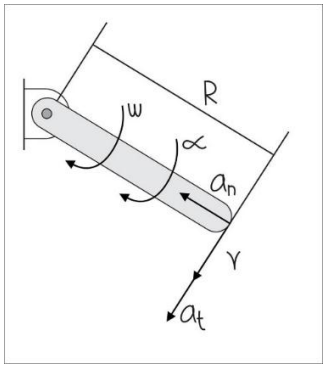
Caracterizando cada uno de los elementos del mecanismo es posible determinar los datos buscados como lo son la velocidad angular y la aceleración angular, las cuales son indispensables en nuestro proyecto ya que son los elementos que se deben buscar en nuestro proyecto, pero para conseguir estos valores en cada instante de tiempo debemos encontrar otros valores que guíen a la consecución de estos datos, por ejemplo, posiciones relativas de un punto con respecto a otro, velocidades de diferentes puntos y aceleraciones también.

El mecanismo que a nosotros nos concierne es el de manivela biela corredera, ya que sobre este fue que se hicieron todas las pruebas del software TRACKER, por lo que es importante realizarle el estudio cinemático a este movimiento, este mecanismo no presenta un movimiento donde su manivela se mueva a velocidad angular constante o aceleración angular constante sino presenta una aceleración angular que no sigue ningún parámetro lo que nos genera dificultad para encontrar en la biela los valores de velocidad angular y aceleración angular para cualquier punto de este solido rígido.

7.1. CINEMÁTICA EN LA MANIVELA

Se analizara el primer elemento del mecanismo, el cual es la manivela, como primera medida se debe aclarar que no se hizo un análisis cinemático que relacione los puntos del extremo de la manivela con respecto a la variación del ángulo de giro, ya que el estudio que se realizo fue un el estudio para cada instante de tiempo, por lo que se tomó la variación de las posiciones, velocidades y aceleraciones con respecto a la variación del tiempo.

Figura 25. Diagrama cinemático de la manivela.



La manivela puede ser tomada como un simple elemento que rota alrededor de un eje fijo, describiendo en ella una circunferencia, ya que esta no presenta una variación en su longitud durante todo su desplazamiento, es decir al rotar 360°.

El análisis de este elemento depende de su variación en su posición con respecto al tiempo, lo que nos dirá, cuanto es el dato de su velocidad angular a lo largo de la manivela, eso significa que para cada punto longitudinal de la manivela se presenta un valor de velocidad angular diferente, aunque el importante para nuestro cálculo es el que se encuentra en su extremo, es decir, donde la manivela y la biela se juntan ya que este valor se tomara como referencia para luego encontrar la velocidad angular en la biela, la ecuación dada es:

$$v = \omega * R \qquad (3)$$

Y para el caso de la aceleración también es i**m**portante encontrarla donde se genera un encuentro entre la manivela y la biela, por la razón antes mencionada, es de mostrar que el análisis cinemático de la aceleración se torna un poco más compleja ya que está se encuentra compuesta por dos componentes la aceleración tangencial y la aceleración normal, la suma de estas dos componentes nos dan la aceleración total, la razón es que son ortogonales y hallamos la aceleración como vector, las ecuaciones que nos definen la aceleración son:

$$\vec{a} = \overrightarrow{a_t} + \overrightarrow{a_n} \qquad (4)$$

$$\overrightarrow{a_t} = \vec{\alpha} * \vec{R} \qquad (5)$$

$$\overrightarrow{a_n} = \vec{\omega} * \left(\vec{\omega} * \vec{R}\right) \qquad (6)$$

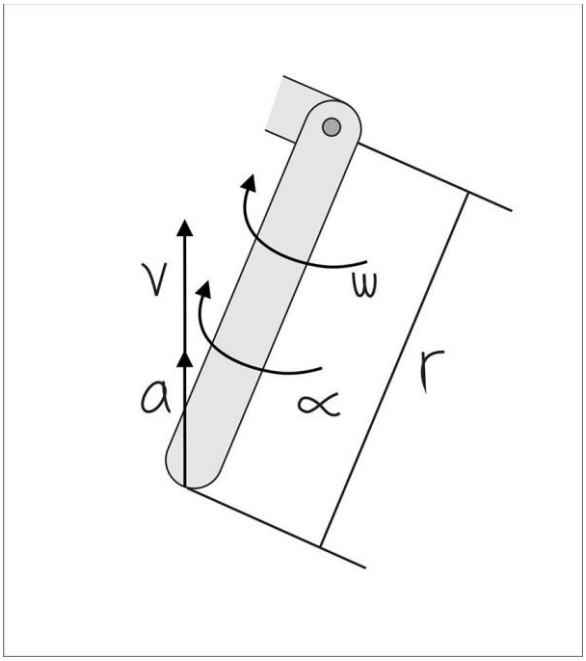
Tabla 3. Datos de manivela.

Manivela					
x [cm]	y [cm]	v_{y} [cm/s]	v_{x} [cm/s]	a_{x} [cm/s^2]	a_{y} [cm/s^2]
16,00	-8,81	0,00	0,00	0,00	0,00
15,60	-9,39	-17,40	-10,40	0,00	0,00
15,30	-9,97	-18,30	-11,30	-25,30	-3,64
14,80	-10,60	-18,10	-12,60	-16,50	13,40
14,40	-11,20	-16,90	-12,20	-16,40	18,40
14,00	-11,70	-17,00	-13,50	-27,50	14,90
13,50	-12,30	-16,10	-14,60	-29,80	16,50
13,00	-12,80	-15,50	-15,00	-29,50	-4,12
12,50	-13,30	-16,50	-16,70	-16,50	-4,72
11,90	-13,90	-16,10	-16,40	4,79	52,00
11,40	-14,40	-13,00	-15,80	14,10	62,40
10,90	-14,80	-11,40	-15,90	-19,60	29,80
10,40	-15,20	-11,50	-16,70	-41,50	-10,50
9,76	-15,50	-11,80	-19,00	-42,00	8,49
9,09	-16,00	-11,40	-19,60	-24,80	-0,54
8,45	-16,30	-11,30	-20,30	0,79	38,90
7,74	-16,70	-9,64	-20,10	-20,30	24,30
7,11	-16,90	-8,49	-20,90	-45,30	14,90
6,34	-17,30	-9,55	-23,70	-41,40	12,30
5,53	-17,60	-7,52	-23,70	-2,85	48,30

7.2. CINEMÁTICA DE LA BIELA

En la biela es donde más se encuentran puntos de interés, porque en un punto se encuentra la intersección entre la manivela y la biela, otro es el punto donde la biela se une con el carretel y por ultimo cualquier punto que compone la biela, porque ese es el objetivo del proyecto que se pueda hallar la velocidad angular y la aceleración angular del solido rígido.

Figura 26. Diagrama cinemático de la biela.

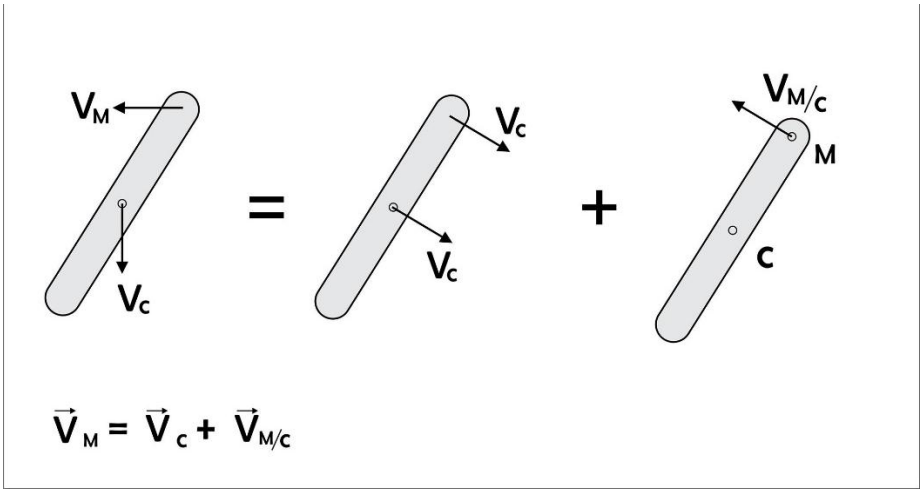


Para lograr estos datos se tiene que hacer un proceso iterativo donde para encontrar los valores de dichas características en un punto cualquiera necesitamos la referencia de los valores conocidos en otro punto, el punto conocido se toma como el de la intersección entre la manivela y la biela ya que estos valores fueron encontrados con el proceso descrito anteriormente.

Luego se debe tener en cuenta los valores de la velocidad y aceleración relativas por lo que estas relacionan dos puntos, entre los que se tienen uno conocido y otro que se desea encontrar.

En las velocidades su análisis vectorial es:

Figura 27. Análisis de velocidades.



$$\overrightarrow{v_m} = \overrightarrow{v_c} + \overrightarrow{vm_{/c}} \qquad (7)$$

Y en las aceleraciones.

$$\overrightarrow{a_m} = \overrightarrow{a_c} + \overrightarrow{am_{/c}} \qquad (8)$$

Tabla 4. Datos punto medio

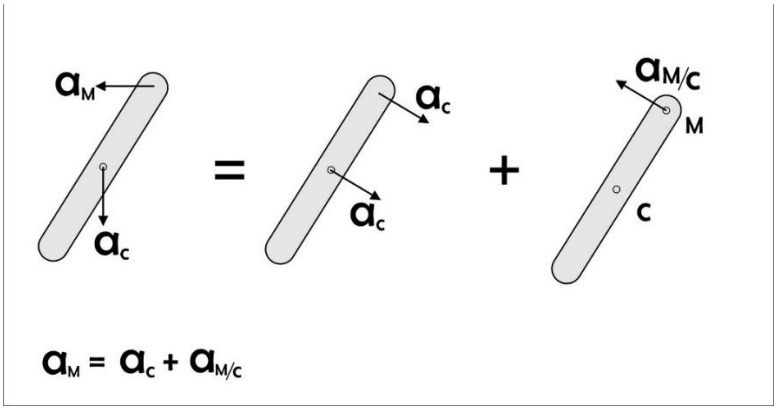
Punto medio					
x	y	a_{x}	v_{x}	a_{y}	v_{y}
8,80952381	-35,48	0,00	0,00	0,00	0,00
8,62667834	-36,20	0,00	-5,71	0,00	-20,13
8,42889996	-36,82	-5,85	-5,79	23,22	-18,73
8,24035567	-37,45	-0,91	-5,98	13,44	-18,87
8,03012262	-38,08	-17,17	-6,01	19,27	-17,92
7,83964812	-38,64	-23,59	-6,83	-2,98	-17,30
7,57441984	-39,23	-15,25	-7,95	-8,00	-18,23
7,30973939	-39,86	15,84	-7,72	31,28	-18,09
7,05964459	-40,44	-12,53	-6,91	44,78	-16,01
6,8487643	-40,93	-14,64	-8,16	30,80	-14,83
6,5157109	-41,43	-24,86	-8,68	2,92	-14,27
6,26972889	-41,88	-17,70	-8,95	14,57	-14,32
5,91855333	-42,38	-0,04	-10,50	29,96	-13,79
5,56923208	-42,80	7,00	-8,94	34,43	-11,94
5,32215766	-43,18	-19,34	-9,33	6,21	-11,49
4,94708862	-43,57	-28,67	-11,03	5,99	-11,64
4,58650531	-43,96	-1,11	-10,89	31,25	-11,15
4,22061289	-44,31	-12,51	-11,14	29,94	-9,56
3,84387984	-44,59	-14,90	-11,70	32,53	-8,87
3,44001645	-44,90	-17,36	-12,21	27,97	-7,85

Datos del punto medio de la biela que en la ilustración 27 se identifica como el punto C, de nuevo se obtuvieron datos de interes como lo son posicion, velocidad y aceleracion en el eje cartesiano. Tambien es necesario realizar una simulación del punto llamado carretel por lo cual tambien se presentaran los datos generados por TRACKER.

Tabla 5. Datos del carretel.

CARRETEL					
x	y	v_{x}	v_{y}	a_{y}	a_{x}
1,39497167	-63,703706	0	0	0	0
1,37774318	-64,3963354	-0,01311048	-19,8569338	0	0
1,39409732	-65,0279801	0,3127926	-20,3020102	16,0912202	5,75452503
1,39860356	-65,7502919	0,23386433	-19,5976975	-9,0455326	2,33944105
1,40969391	-66,3349655	0,49867232	-19,7979432	14,7939381	-0,13180728
1,43186039	-67,0706318	0,30921379	-19,8664425	23,8974927	-4,14218667
1,43031561	-67,6598737	0,07918718	-17,305632	42,3179812	-13,9565591
1,43714145	-68,2247576	-0,44245612	-17,2143831	-0,19208399	3,20263285
1,40080788	-68,807914	-0,05750365	-17,4538348	17,9232141	-6,97518344
1,43330648	-69,3887671	-0,33343405	-16,1512164	19,9626818	2,38842829
1,3785709	-69,8850509	-0,57214835	-15,6548829	14,7205717	6,42427753
1,39514948	-70,4328032	0,5761138	-15,6780805	16,6748368	6,35551315
1,41699237	-70,930634	-0,16180789	-14,3075568	30,8669848	-2,05427364
1,38435838	-71,3869851	0,04780882	-13,5658695	23,1246923	0,44429975
1,42018078	-71,8353522	0,37735636	-12,8982713	35,292775	-6,60724433
1,40952457	-72,2471806	-0,68621669	-11,2324387	28,9320007	-0,10839428
1,37441647	-72,5844521	0,18770818	-10,6509865	19,8468284	-1,80775735
1,42204297	-72,957503	-0,17142199	-10,3425373	21,8349595	-4,63120107
1,3629842	-73,2742038	-0,88239328	-8,95434684	28,8348148	-1,7559308
1,36319549	-73,5546752	0,17047647	-8,38827656	15,5799751	15,3427192

Figura 28. Análisis de aceleraciones.



Al realizar este proceso se logra determinar las velocidades y aceleraciones de un punto intermedio de la biela, luego se debe realizar el estudio vectorial para lograr determinar con esos datos la velocidad angular y la aceleración angular. Las ecuaciones usadas para determinar esos datos son las siguientes, desarrollando la ecuación de la velocidad relativa, obtenemos.

$$\overrightarrow{v_c} - \overrightarrow{v_m} = \overrightarrow{rc_{/m}} * \overrightarrow{\omega c_{/m}} \qquad (9)$$

$$\overrightarrow{\omega c_{/m}} = \frac{\overrightarrow{v_c} - \overrightarrow{v_m}}{\overrightarrow{rc_{/m}}} \qquad (10)$$

Es así como se determinan la velocidad angular de un punto cercano al centro de la biela, aunque cabe aclarar que es posible hacerlo para cualquier punto que en ella la compone.

Durante el cálculo de la aceleración angular se debe tener en cuenta, que de igual forma que en la velocidad angular, la ecuación donde se relacionan dos puntos de la biela, tenga un valor conocido de la aceleración en un punto y otro por hallar, y esta ecuación es la de la aceleración relativa, la cual despejándola obtenemos lo siguiente:

$$\overrightarrow{ac_{/m}} = \overrightarrow{a_c} - \overrightarrow{a_m} \qquad (11)$$

$$\overrightarrow{ac_{/m}} = \left(rc_{/m_x} * \alpha c_{/m}\right) j + \left(rc_{/m_y} * \alpha c_{/m}\right) i + \left(rc_{/m_x} * \omega c_{/m}{}^2\right) i + \left(rc_{/m_y} * \omega c_{/m}{}^2\right) j \quad (12)$$

Tomando la aceleración en el eje x se obtiene.

$$a_{c_x} - a_{m_x} = \left(rc_{/m_y} * \alpha c_{/m}\right) + \left(rc_{/m_x} * \omega c_{/m}{}^2\right) \qquad (13)$$

Y luego despejando la aceleración angular se obtiene.

$$\alpha_{c/m} = \frac{a_{c_x} - a_{m_x} - (r_{c/m_x} * \omega_{c/m}{}^2)}{r_{c/m_y}} \qquad (14)$$

Tabla 6. Datos de resultados del punto medio de la biela.

RESULTADOS					
R	Vc-Vm	Wc/m [rad/s]	(a_p - a_m)i	r_x*w^2	alpha [rad/s^2]
27,6186289	0,00	0	0	0	0
27,702602	5,43	-0,195926107	0	0	0
27,7137808	5,52	-0,19933536	19,45	-0,27302031	-0,73456527
27,6396394	6,67	-0,241159749	15,59	-0,38149595	-0,59485127
27,6217185	6,27	-0,227163919	-0,77	-0,32870762	0,016267899
27,6401632	6,67	-0,241453687	3,91	-0,35914779	-0,158257888
27,5753553	6,99	-0,253378948	14,55	-0,38042753	-0,554411687
27,6525276	7,73	-0,279530385	45,34	-0,44462124	-1,691885628
27,6778704	9,80	-0,354073289	3,97	-0,6820459	-0,171396906
27,4965797	8,34	-0,303334295	-19,43	-0,46477276	0,701664756
27,4649785	7,23	-0,263271181	-38,96	-0,33853845	1,429098882
27,4735726	7,54	-0,274314681	1,90	-0,34842116	-0,083202391
27,5494328	6,61	-0,239755362	41,46	-0,25760536	-1,534632707
27,6198674	10,06	-0,364169978	49,00	-0,55577869	-1,815111805
27,4389657	10,27	-0,374343596	5,46	-0,52799953	-0,220374683
27,4903503	9,28	-0,337411504	-29,46	-0,39879428	1,065867662
27,4373715	9,33	-0,340067044	19,19	-0,36468777	-0,717468649
27,5620356	9,82	-0,356399118	32,79	-0,36701091	-1,209518313
27,407011	12,01	-0,438374431	26,50	-0,47968476	-0,988621051
27,3816041	11,50	-0,419872443	-14,51	-0,3684492	0,518000257

Estos son los datos de resultados, donde fueron aplicadas las ecuaciones (10) y (14) para encontrar datos de la velocidad angular y la aceleración angular.

8. RESULTADOS

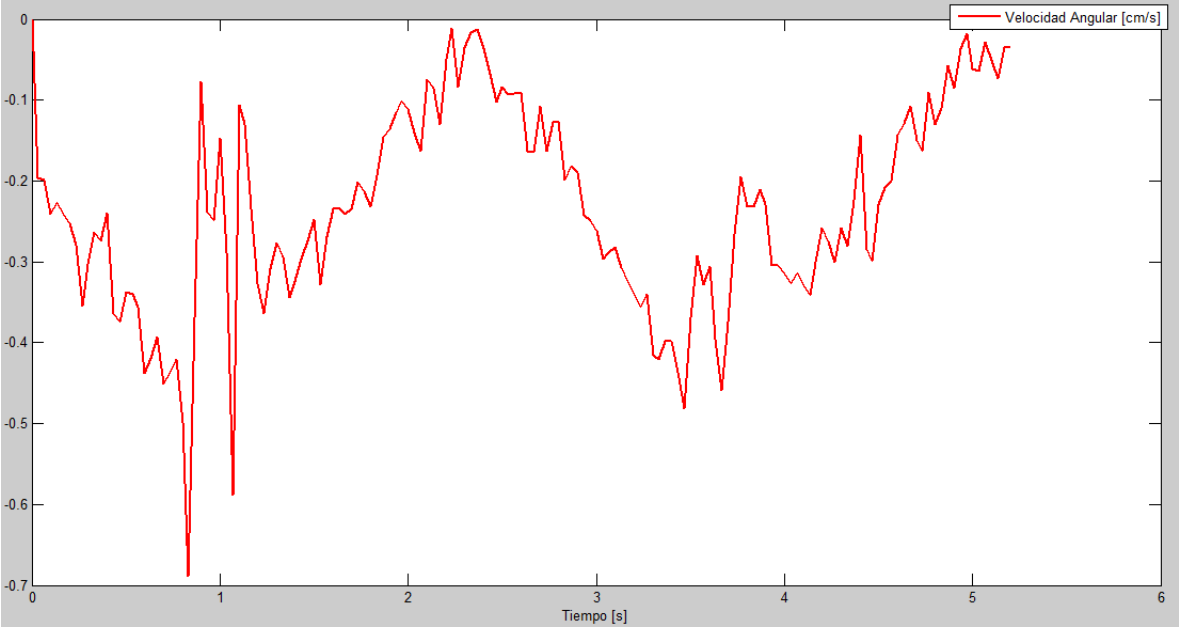
En el presente capítulo se exponen los resultados obtenidos del análisis de un video del movimiento de la biela del mecanismo de prueba, un mecanismo manivela-biela-corredera, el cual se encuentra en el laboratorio de automatización industrial de la UIS y de un video del mismo mecanismo simulado en Solid Works.

8.1 MECANISMO REAL

Para el mecanismo real se grabó un video de aproximadamente 5 segundos de duración en el cual se obtuvieron 159 datos con un intervalo de tiempo de 0,033 segundos.

8.1.1 Velocidad angular El primer dato calculado a partir del análisis del programa TRACKER fue la velocidad angular de la biela a continuación se muestran los resultados obtenidos mediante las formulas planteadas en el capítulo 7.

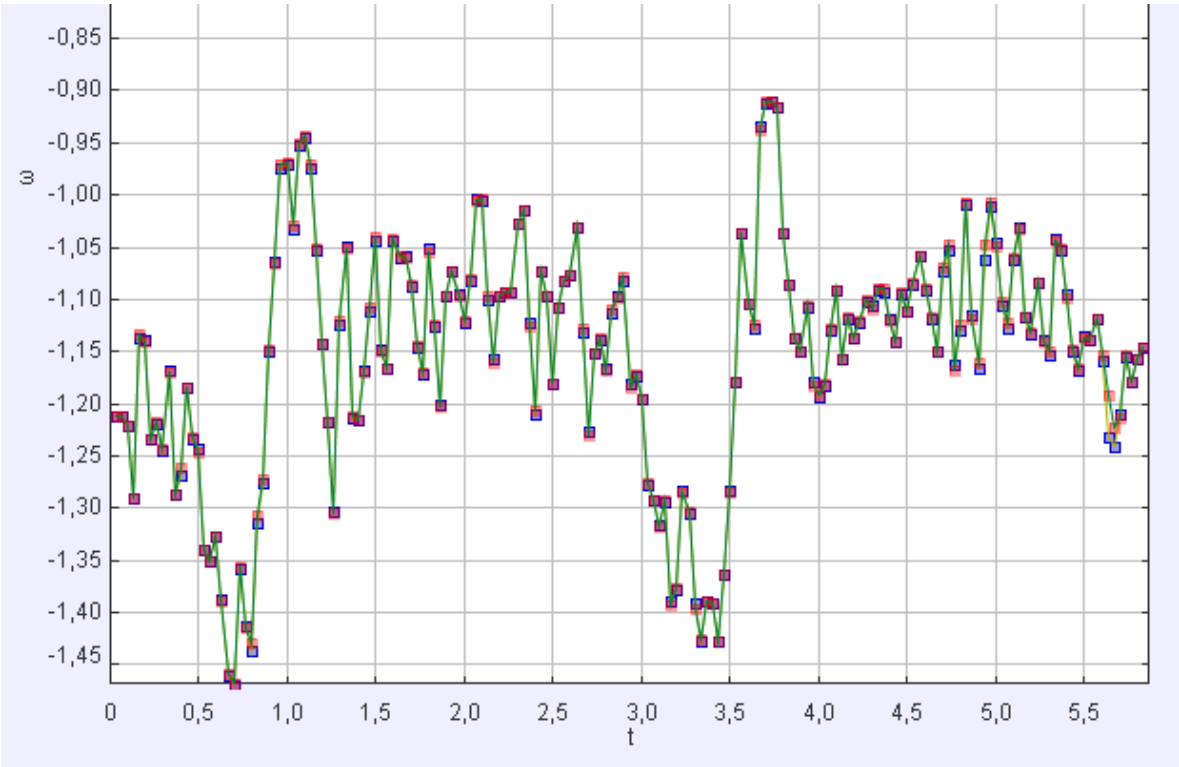
Figura 29. Velocidad angular de la biela respecto al tiempo.



Se observa en la gráfica que el comportamiento de la velocidad es variable, esto es debido a que la velocidad angular de la manivela no es constante ya que el motor varía la corriente que consume durante su giro; y también vemos que se presentan algunos picos debido a posibles errores.

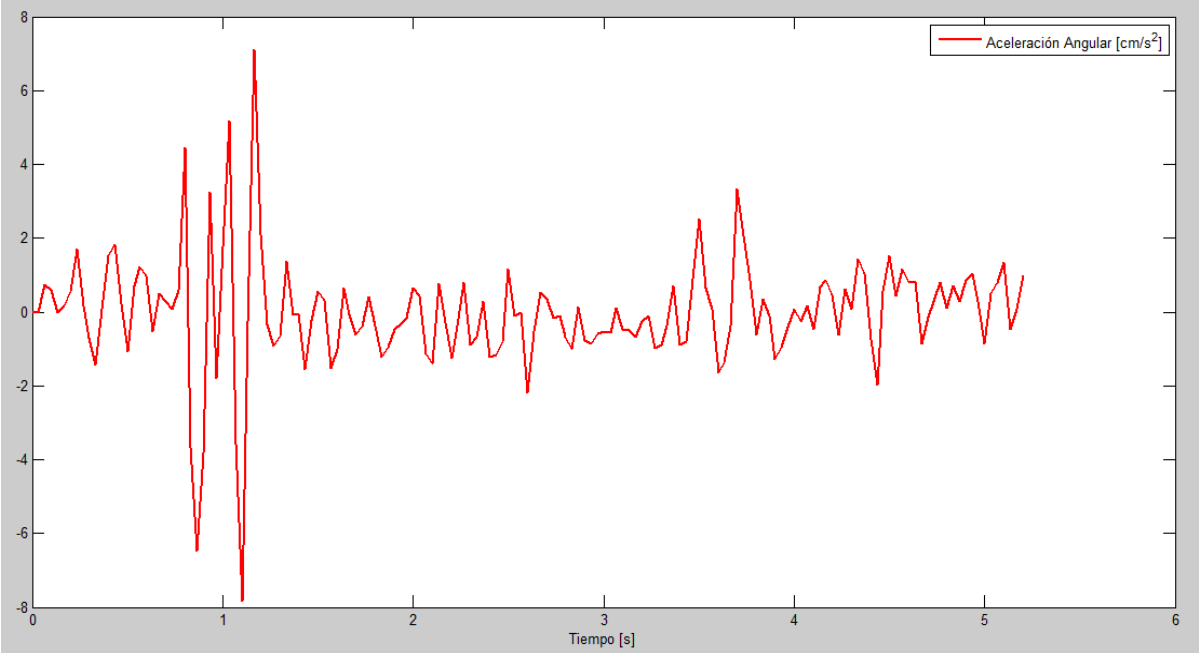
En la gráfica 30 tomada directamente de TRACKER podemos ver que la velocidad angular de la manivela no es constante sino que varía sin marcar una tendencia.

Figura 30. Velocidad angular de la manivela respecto al tiempo.



8.1.2 Aceleración angular También se calculó la aceleración angular y podemos observar los resultados en la gráfica 31. Los datos de aceleración angular no presentan una tendencia fija en su comportamiento, sin embargo su utilidad se comprobará más adelante en este mismo capítulo.

Figura 31. Aceleración angular de la biela respecto al tiempo.

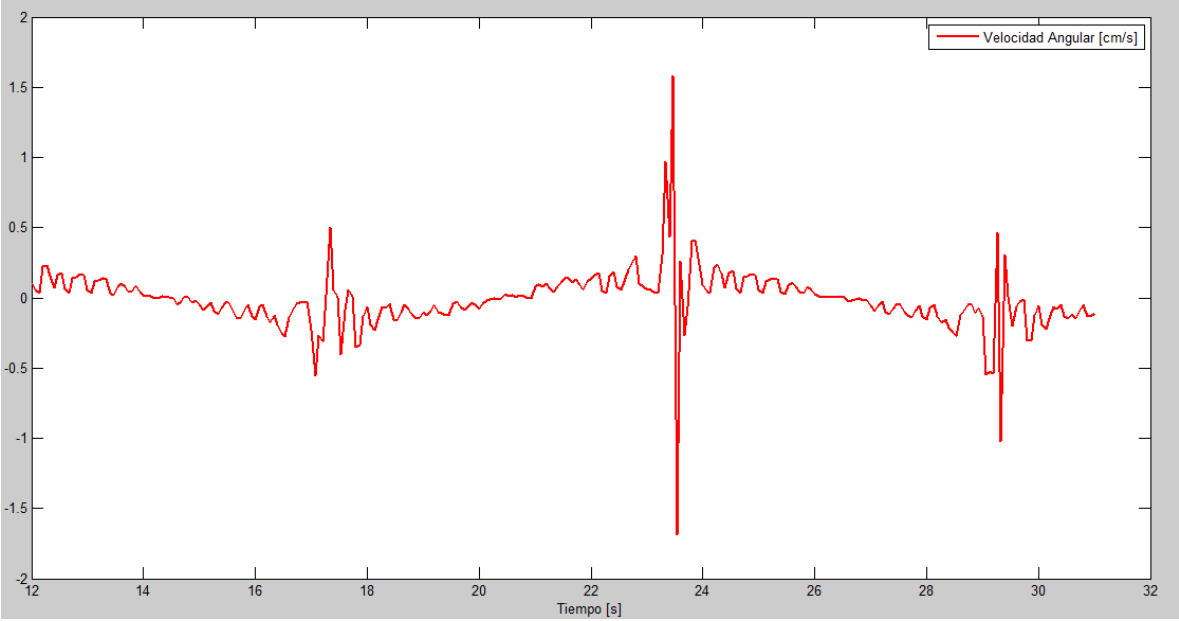


8.2 SIMULACIÓN EN SOLID WORKS

Debido a que el mecanismo real no tiene una velocidad angular constante de entrada, lo cual hace que el movimiento tenga cambios repentinos e impredecibles de velocidad y aceleración, también se realizó una simulación del mecanismo en Solid Works para ver cómo sería el análisis si se tiene una velocidad de la manivela constante, para esto se grabó un video de aproximadamente 19 segundos de duración en el cual se obtuvieron 286 datos con un intervalo de tiempo de 0,066 segundos.

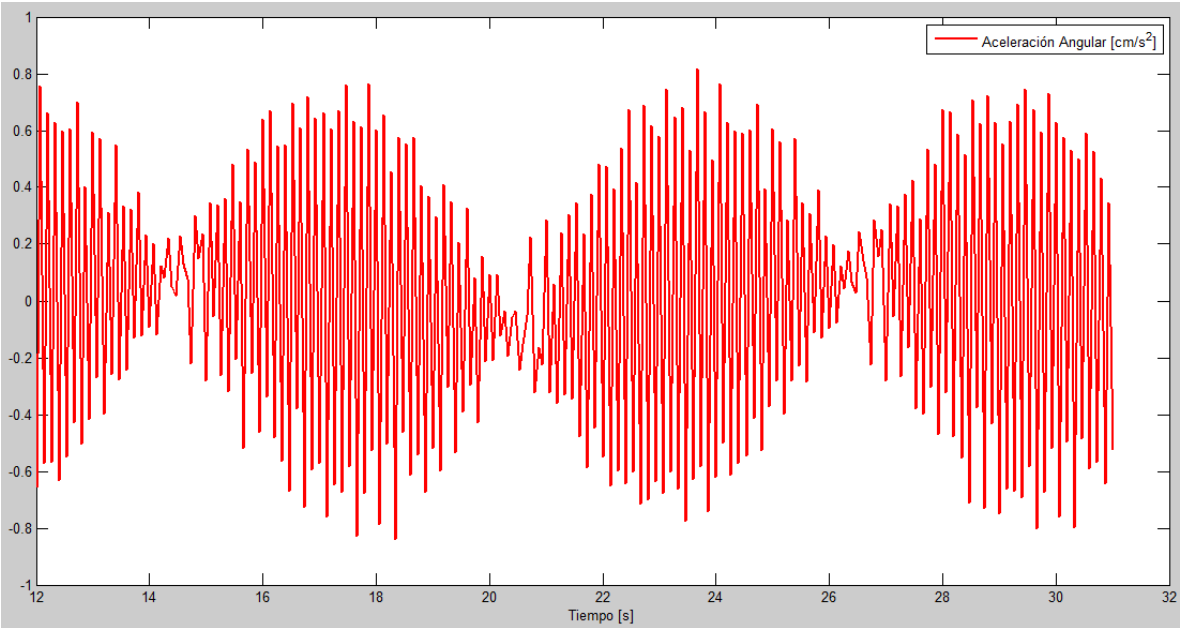
8.2.1 Velocidad angular En la gráfica 32 podemos ver los resultados de la velocidad angular respecto al tiempo, ya no se presentan cambios tan bruscos de la velocidad sin embargo en ciertos puntos de la gráfica hay valores demasiado alejados del comportamiento que presenta la curva debido a errores en el video.

Figura 32. Velocidad angular de la biela de la simulación de SolidWorks.



8.2.2 Aceleración angular En la gráfica 33 observamos los valores calculados de la aceleración angular respecto del tiempo; en los instantes de tiempo en que en el mecanismo la manivela está en posición horizontal es cuando la aceleración angular de la biela tiene amplitud cercana a 0 y en los instantes de tiempo en que la manivela está en posición vertical la amplitud de la aceleración angular de la biela es máxima pero a medida que cambian estos valores la aceleración va cambiando de sentido positivo a negativo.

Figura 33. Aceleración angular de la biela respecto al tiempo en la simulación de SolidWorks.



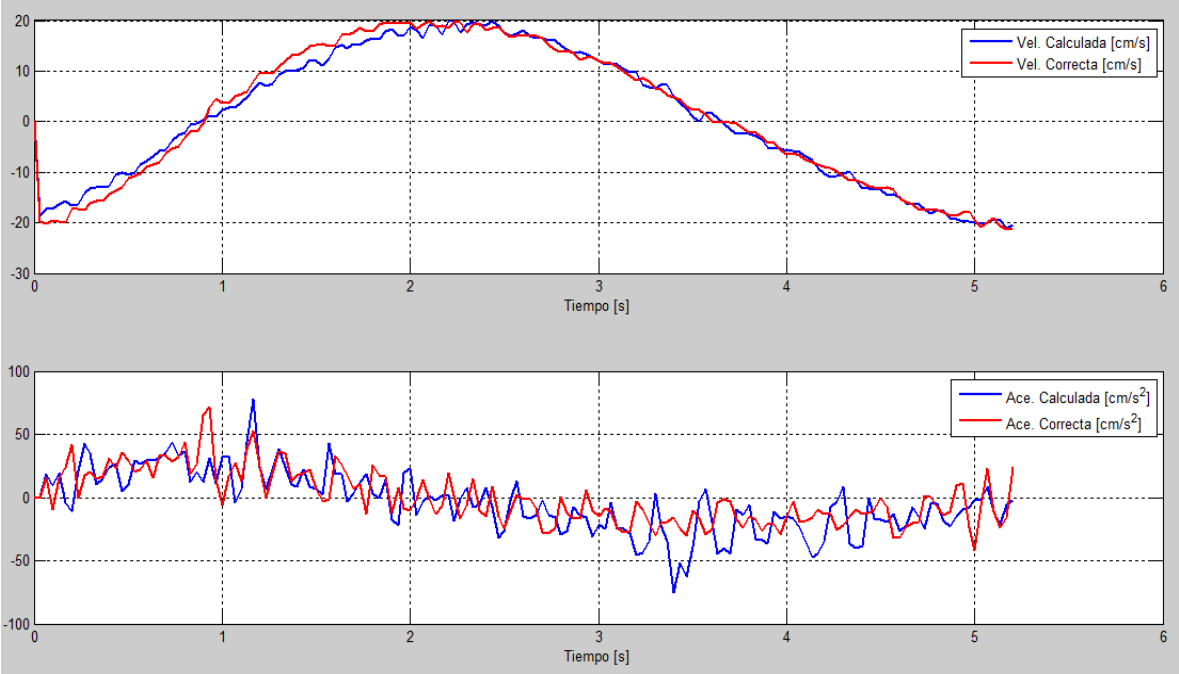
8.3 VALIDACION DE RESULTADOS

Puesto que los datos presentados son datos obtenidos a partir de análisis reales, es difícil tener la certeza de cómo deberían ser los resultados para poder compararlos, obtener el error y así comprobar si los resultados obtenidos son válidos; para poder tener un punto de referencia con el cual probar la validez de los datos se siguió el siguiente procedimiento: Con los datos de velocidad y aceleración angular junto con los datos de velocidad y aceleración lineal que unen la biela y la manivela se calculó la velocidad y aceleración lineal del punto de la biela que se encuentra conectado a la corredera, la cual tiene solo movimiento en el eje “y”; Estos datos se compararon con los datos de velocidad y aceleración lineal del punto de la corredera obtenidos de TRACKER, debido a que estos últimos se sacaron directamente de TRACKER y son datos confiables se asumirán

como los valores correctos los cuales deberían ser iguales a los calculados con los resultados obtenidos anteriormente.

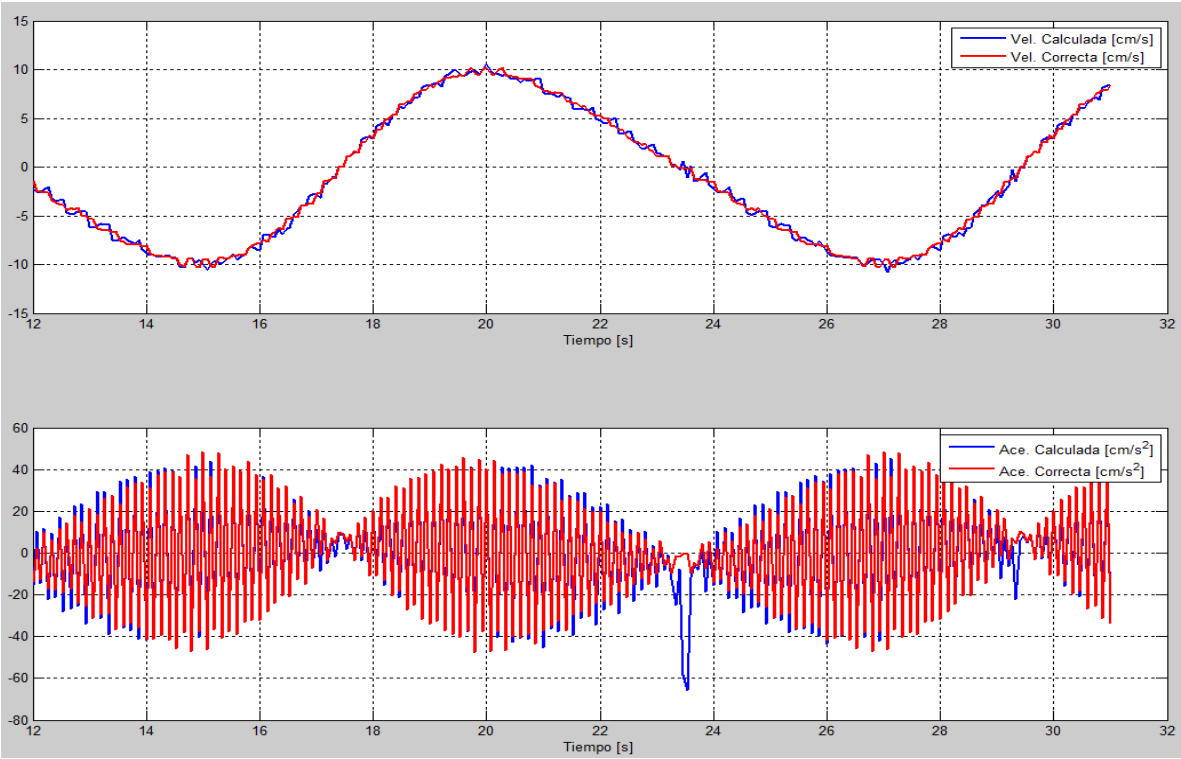
A continuación se presentan las gráficas obtenidas donde se compara la velocidad y aceleración correcta con la velocidad y aceleración calculada del punto de la corredera.

Figura 34. Comparación de velocidades y aceleraciones calculadas y de TRACKER.



De igual forma se obtuvieron las gráficas de velocidad y aceleración lineal del punto de la corredera para la simulación de Solid Works

Figura 35. Comparación de velocidades y aceleraciones calculadas y de TRACKER, simulación en SolidWorks.



8.3.1 Error cuadrático medio y coeficiente de Pearson Para determinar la similitud entre los datos de la gráfica 34 y la gráfica 35 se calcularon los errores, el error cuadrático medio y se determinó el coeficiente de Pearson.

El error cuadrático medio es un criterio estadístico para estimar el promedio de los errores al cuadrado, se considera el cuadrado de ese error para evitar que las diferencias positivas se compensen con las negativas, siendo lo ideal tener un error cuadrático medio que tienda a 0.

A continuación se muestra la fórmula aplicada para hallar el error cuadrático medio, y el porcentaje de error de la velocidad (que de igual manera se empleó para la aceleración) es la siguiente:

$$Error = A_{teorica} - A_{exp}$$

$$ECM = \frac{\sum Error^2}{t_{final} - t_{inicial}}$$

Además se calculó el porcentaje de error:

$$\%Error = 100 * \frac{\sum Error^2}{\sum A_{teorica}{}^2}$$

El coeficiente de Pearson (r) proporciona una medida numérica para medir la correlación entre dos señales; si el valor de este coeficiente tiende a 1 significa que existe una alta relación lineal lo cual indica que una de las señales es múltiplo de la otra, por el contrario, si tiende a 0 significa que no existe relación lineal y una señal no puede expresarse en términos de la otra.

Lo coeficientes de Pearson para las señales de velocidad y aceleración fueron hallados usando Matlab.

8.3.1.1 Mecanismo real Aplicando las formulas anteriores se obtuvieron los siguientes resultados:

$$ECM_{Velocidad} = 2{,}5186$$

$$ECM_{Aceleracion} = 385{,}7119$$

$$\%Error_{Velocidad} = \ 1{,}3434\ \%$$

$$\%Error_{Aceleracion} = 86{,}6012\%$$

Coeficiente de Pearson:

$$r_{vel} = 0{,}9940$$

$$r_{ace} = 0{,}6485$$

8.3.1.2 Simulación en Solid Works Para los datos obtenidos de Solid Works los resultados fueron los siguientes:

$$ECM_{Velocidad} = 0{,}2068$$

$$ECM_{Aceleracion} = 59{,}4021$$

$$\%Error_{Velocidad} = \ 0{,}4587\ \%$$

$$\%Error_{Aceleracion} = 6{,}9174\%$$

Coeficiente de Pearson:

$$r_{vel} = 0{,}9976$$

$$r_{ace} = 0{,}9657$$

8.3.2 Análisis del error De los valores de los errores calculados anteriormente, podemos observar que para los datos de la velocidad se tienen errores bajos y un coeficiente de Pearson cercano a 1 lo cual indica que hay una alta relación lineal, aunque para los datos de la aceleración dieron errores considerablemente altos para el mecanismo real.

Sin embargo para los datos de la simulación de Solid Works los errores fueron significativamente más bajos que los del video del mecanismo real.

Los errores que se presentan se deben principalmente a:

- Movimientos de la cámara al momento de grabar el video,
- El plano de captura del video no es completamente paralelo al plano donde se encuentra el mecanismo,
- Se analizan datos discretos y no continuos
- Al utilizar valores con errores en las fórmulas de cinemática se propagan los errores.

Varios de los errores mencionados anteriormente son inherentes a la captura de video de mecanismos reales, es por esto que al hacer la simulación en Solid Works se redujo significativamente los errores.

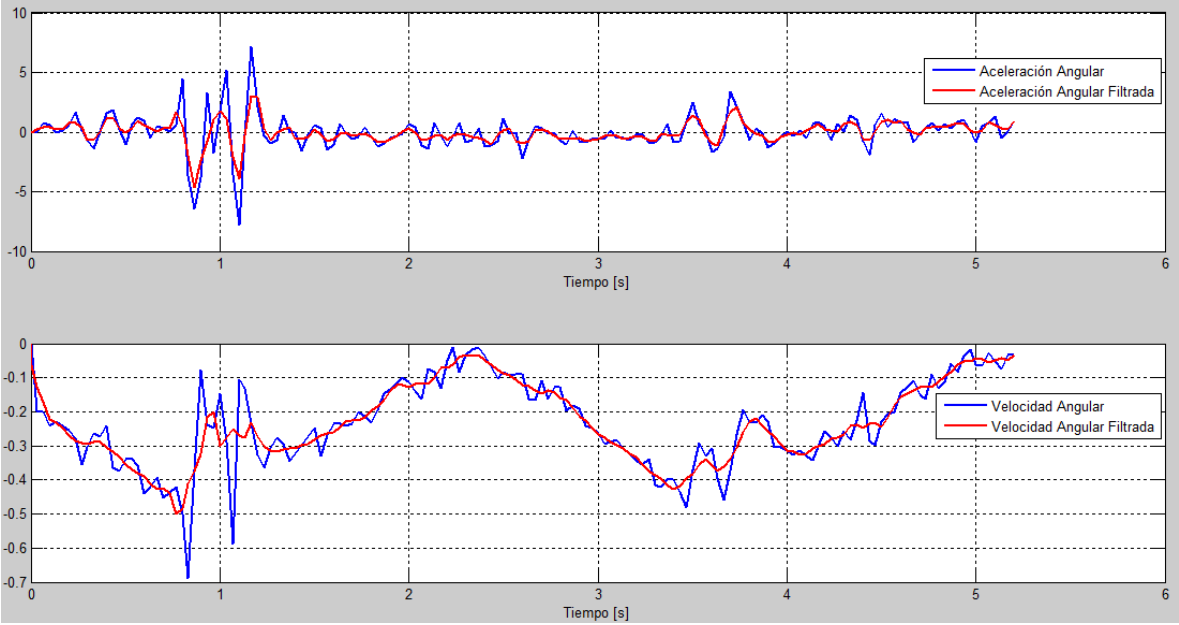
8.4 FILTRADO DE RESULTADOS

Puesto que los errores de los resultados de la aceleración fueron demasiado altos, se procedió a hacer un tratamiento a la señal de la aceleración angular y las aceleraciones lineales calculadas y las que se obtuvieron por Tracker, el filtrado también se realizó a los datos de las velocidades con la intención de disminuir el error.

Para disminuir el error se utilizó un filtro de señales pasa-bajas, el cual atenúa las frecuencias altas bajas, para ello, se utilizó una función de Matlab llamada “Smooth”, la cual tiene como entrada los datos de velocidad y aceleración y devuelve los correspondientes datos filtrados.

A continuación se muestran las gráficas de la velocidad angular y aceleración angular de la biela después de aplicarles el filtro en Matlab.

Figura 36. Comparación entre las velocidades y aceleraciones angulares filtradas y sin filtrar.



Además se calcularon los errores después de filtrar los datos de las velocidades y aceleraciones lineales del punto que une la corredera con la biela.

Figura 37. Comparación de las velocidades lineales filtradas y de los errores que estas generan.

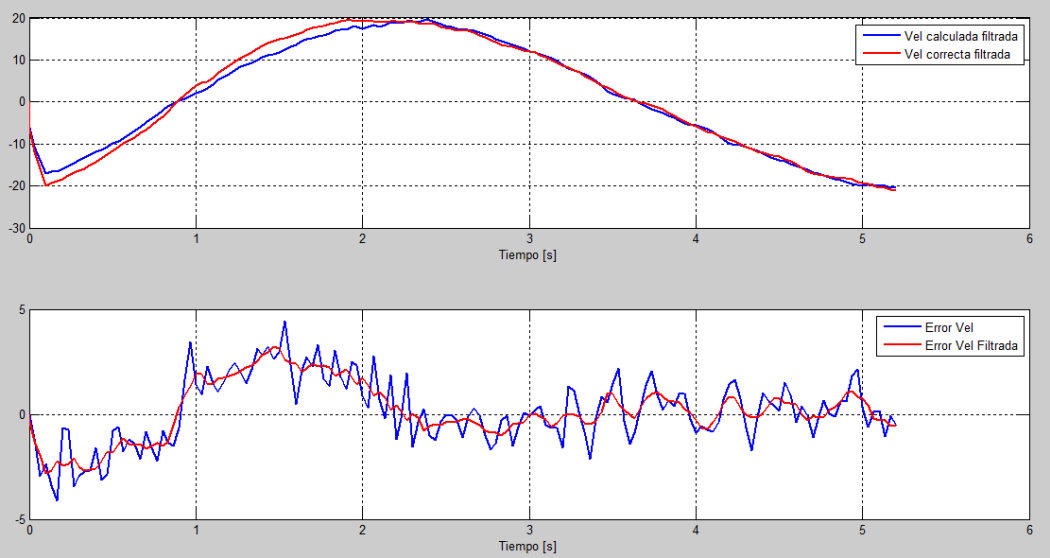
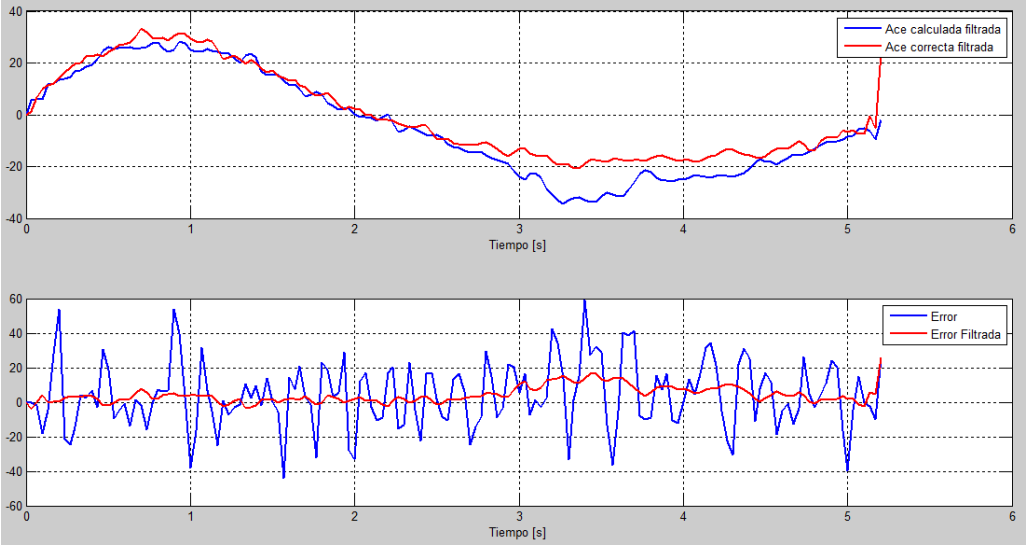


Figura 38. Comparación de las aceleraciones lineales filtradas y de los errores que estas generan.



En las gráficas podemos observar que se disminuye considerablemente el error al aplicar este filtro a los datos del mecanismo real, a continuación se presentan los

valores numéricos de los errores y el coeficiente de Pearson calculados para los datos filtrados:

$$ECM_{Velocidad} = 1{,}7983$$

$$ECM_{Aceleracion} = 40{,}3563$$

$$\%Error_{Velocidad} = \ 0{,}9751\ \%$$

$$\%Error_{Aceleracion} = 14{,}0966\%$$

Coeficiente de Pearson:

$$r_{vel} = 0{,}9959$$

$$r_{ace} = 0{,}9749$$

Se puede observar una notoria mejora en todos los valores calculados; se disminuyó el error cuadrático medio así como el porcentaje de error y además el coeficiente de correlación lineal de Pearson se acercó considerablemente a 1, lo cual es deseado.

9. CONCLUSIONES

- Se concluye que es posible obtener la información necesaria a partir de un video, para poder realizar un análisis cinemático aproximado de un movimiento de la realidad, tomando en cuenta las ecuaciones que describen el movimiento plano general de un sólido rígido.

- El modelado de los movimientos a partir del análisis de un video, es una herramienta que permite comparar la realidad con la teoría de la cinemática del solido rígido y además ayuda a tener una percepción más clara del movimiento de diferentes tipos de mecanismos.

- Por medio de este proyecto se logró cooperar con el desarrollo de una herramienta que le permite a los estudiantes de ingeniería mecánica afianzar conocimientos y competencias en el área de dinámica, contribuyendo así con el propósito de la universidad Industrial de Santander que es formar integralmente a los estudiantes y con habilidades para aplicar el conocimiento de las ciencias y la ingeniería.

- Se logró obtener el desarrollo de un software de modelamiento computacional, el cual permite realizar un análisis cinemático de un solio rígido mediante la obtención de video, que contenga el fenómeno físico que se desee estudiar, este software presenta las siguientes características:

    - Realizando el estudio cinemático permite la obtención de datos relevantes y característicos en el análisis físico de un sólido rígido en dos dimensiones como lo son la posición, la velocidad angular y la aceleración angular.

- Este software mediante la detección automática por medio de identificador de colores previamente dispuestos en el mecanismo, le da una guía al programa de los elementos de referencia necesarios para desarrollar correctamente el algoritmo de cálculo de cada dato físico.
- Durante la elaboración del programa se logró realizar un análisis de 2 elementos del solido rígido, fue elaborado en un lenguaje enfocado a objetos, JAVA, en la plataforma ECLIPSE usando videos de 30 fotogramas por segundo.

❖ Se diseñó el procedimiento de la creación del video del movimiento del solido rígido, añadiéndole identificadores que le permite tener una detección de colores automático acertado y la adecuada forma de crear el video permitiendo así tener resultados muy cercanos a lo esperado.

10. RECOMENDACIONES

Para realizar una adecuada mejora para un próximo proyecto se recomienda crear una aplicación en ANDROID, para que se facilite su uso al poder ser usado como una aplicación de un Smartphone, lo que resultaría una ayuda al usuario de *KINE-UIS* , solo necesitando de un celular que contenga la aplicación, y desde la misma filmar el video del mecanismo dinámico al cual se le desea realizar el estudio.

También se recomienda que al momento de crear el video se use un tamaño adecuado como lo es 760 x 1080 que corresponden al alto del fotograma y al ancho del fotograma, también se recomienda una velocidad de 30 fotogramas por segundo, ya que si se aumenta de tamaño o de velocidad de fotogramas, podría producir un análisis muy lento por parte de *KINE-UIS*, tampoco se aconseja que el video sea creado con calidad menor debido a la disminución en la calidad, lo que produce reducción en la resolución aumentando la posibilidad de perder los identificadores en la detección automática y en el seguimiento de la masa puntual.

BIBLIOGRAFÍA

BEER, F. y JONHSON, R. Mecánica vectorial para ingenieros. Estática (6ª ed.). México: McGraw-Hill. 1997.

BROWN, Douglas. Video Modeling: Combining Dynamic Model Simulations with Traditional Video Analysis [online]. Avalible from Internet: <http://cabrillo.edu/~dbrown/tracker/Video%20Modeling.pdf>.

PORTAL DE OPEN SOURCE PHYSICS. [online]. Avalible from Internet: http://www.opensourcephysics.org/

PORTAL DE TRACKER, Video Analysis and Modeling Tool, [online]. Avalible from Internet: http://physlets.org/tracker/

WEE, Loo Kang. Using Tracker as a pedagogical tool for understanding projectile motion [online]. Avalible from Internet: <http://iopscience.iop.org/article/10.1088/0031-9120/47/4/448/meta>.

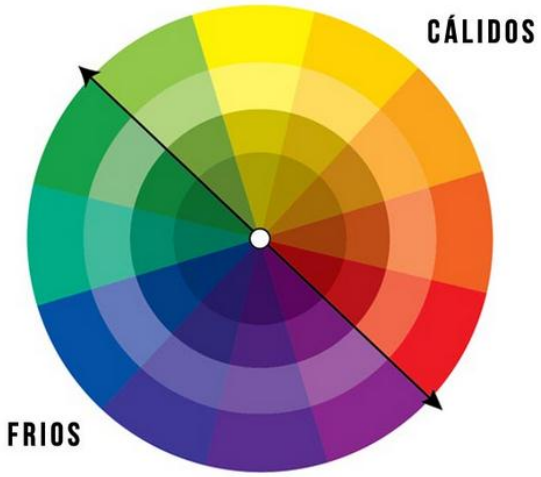
ANEXOS

Anexo A. Fundamentación teórica del procesamiento de imágenes.

Color:

Sabiendo que el color simplemente es una percepción que tienen algunos seres vivos de identificar algunas longitudes de onda, las cuales son traducidas en nuestro cerebro dándonos una idea de color en lo que observamos. Desde siempre se ha requerido tener una estandarización de colores y es por esto que se han intentado dar nombres a estos pero es imposible dar nombre a cada una de los posibles resultados de los colores. Es por esto que se comenzó a hablar de colores cálidos y colores fríos y estos en sus diferentes matices.

Ilustración 1. Círculo cromático dividido en dos regiones.



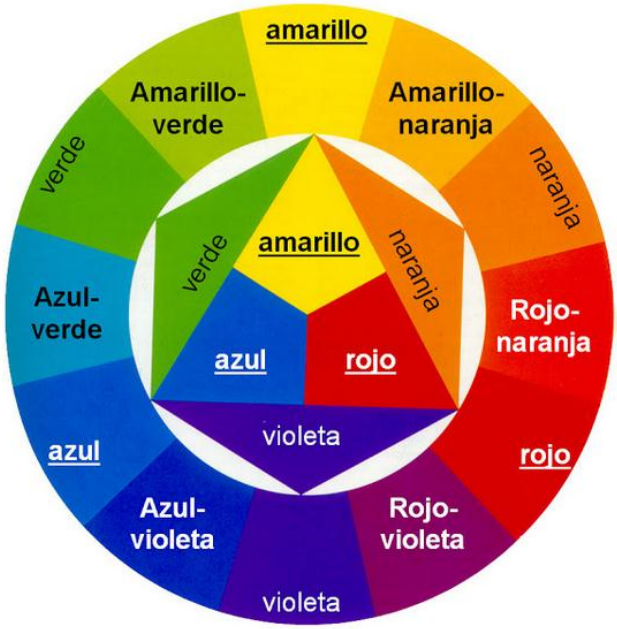
Fuente: La psicología del color.

CLASIFICACION DE LOS COLORES

La mayoría de las clasificaciones parten de tres colores llamados primarios de los cuales se desarrollan el resto, estos son, el color rojo, amarillo y azul, para esto se

creó el circulo cromático, el cual es una sencilla representación de las posibles combinaciones de los colores primarios para formar nuevos, este es un círculo dividido en 12 partes en este se coloca el amarillo en la parte superior, este siendo el color más cálido y en su oposición, es decir en su extremo el color violeta este por ser el color más frio, los colores opuestos se colocan en el círculo cromático en el diámetro opuesto.

Ilustración 2. Circulo cromático con colores secundarios.



Fuente: Pintura Selbex

Los colores secundarios son los creados por la combinación de los primarios, estos son:

ROJO + AMARILLO = NARANJA

ROJO + AZUL = VIOLETA

AMARILLO + AZUL = VERDE

Los colores terciarios son la combinación entre un color primario con los secundarios que son adyacentes, esto es para lograr un aclaramiento o un oscurecimiento a un determinado color, esto da como resultado una matización de los colores.

Los colores análogos son aquellos que se encuentran al lado de un color en el círculo cromático, lo cual hace que sean similares en algunos tintes, por ejemplo el rojo con el naranja.

Los colores opuestos o complementarios son aquellos que están ubicados al otro extremo del circulo cromático, esto es para dar una referencia del color opuesto con respecto a otro color, el color opuesto a un color primario es aquel color secundario que es formado por los otros dos colores primarios. Por ejemplo el opuesto al color rojo es el color verde.

Ilustración 3. Colores opuestos.

Fuente: Star blue comics.

Pero esto no era suficiente para establecer la diferente variedad infinita de colores que había, por lo que se postularon modelos que intentan una estandarización de

los colores, para lograr identificarlos en cualquier lugar del planeta, uno de estos modelos, el cual nos interesa en este trabajo es el RGB.

SEGMENTACION DEL UMBRAL O THRESHOLDING

Esta segmentación trata de hacer una búsqueda en cada pixel que compone la imagen para realizar una comparación de cada pixel con un parámetro puesto en el umbral que nos permite clasificar los pixeles en “pixeles de primer grado” y “pixeles de segundo grado”, los cuales nos permiten hacer una discriminación entre los pixeles de una imagen que nos permite realizar un trabajo con los pixeles que sean relevantes.

Ilustración 4. Segmentación de los contornos en un cuadro.

Fuente: Mathworks.

RUIDO

El ruido es un efecto que hace que el contenido de pixeles varíen un poco, estos efectos pueden ser posibles debido a circuitos de la cámara o a un determinado scanner, los cuales pueden generar una variación en el brillo de la imagen, estos también pueden darse por la segmentación del umbral, donde este umbral puede determinar el paso de unos pixeles con determinado brillo y así podrían pasar pixeles con un brillo alto o bajo al deseado.

Ilustración 5. Imagen original vs imagen con ruido digital.

Fuente: Digitalfotored.

SUAVIZAMIENTO O SMOOTHING

Es el proceso que se realiza para realizar una reducción del ruido en la imagen, para esto se hace un análisis de todos los pixeles con sus respectivos vecinos, realizando una comparación entre el pixel central con todos sus vecinos, si la intensidad del pixel central se encuentra entre el rango de los pixeles vecinos, se deja tal y como está el pixel, en cambio si el pixel central tiene una intensidad que sobrepasa la intensidad de sus vecinos lo que se hace es tomar la mayor intensidad de los pixeles vecinos y asignárselo como propia, de igual forma si esta con una intensidad muy baja a comparación a la de sus vecinos, solo que en este se le asigna la intensidad del pixel con más baja intensidad de sus vecinos y así lograr una reducción en el ruido, este proceso es logrado de forma sencilla con OpenCV, gracias a la función cvSmooth.

DETECCION DE BORDES

Este es un elemento muy importante a la hora de realizar la detección de figuras geométricas ya que nos permite identificar en una imagen los contornos de los objetos que la componen o de la segmentación de la región que en ella se encuentran, la detección de bordes lo que hace es identificar esos puntos donde

se presenta un cambio abrupto de brillo es decir que exista una discontinuidad en la imagen, la cual nos puede dar un punto de referencia de donde comienza o termina un objeto, esto se hace con la acción de la primera y segunda derivada, para así lograr encontrar los puntos deseados, el resultado de aplicar esta técnicas es la reducción de la información que se tiene, ya que pasamos de tener datos de cada pixel de la imagen a solo unos cuantos, el cual está compuesto por el borde de los objetos que en ella se encuentran.

Existe un algoritmo de detección de bordes, el cual lleva por nombre detección de bordes Canny, el nombre fue dado por la persona que creo este algoritmo Jhon Canny, este algoritmo trabaja con máscaras las cuales representan una serie de aproximaciones de diferencias finitas, lo que hace es tener dos umbrales uno máximo y otro mínimo de tal forma que si sobrepasa el umbral máximo el pixel es tomado como borde si está por debajo del mínimo se descarta y si se encuentra entre los dos rangos el pixel solo será aceptado como borde si este se encuentra junto a otro pixel que supera el umbral máximo, en este proceso por lo tanto se le deben dar dos valores que caracterizan el trabajo y son el del umbral máximo y mínimo, si el umbral máximo que se da es muy alto, es posible que varios bordes no sean encontrados; de la misma forma si se coloca un valor muy bajo existirá mucho ruido en la imagen de resultado.

Este método es considerado por ser uno de los más eficientes para realizar la detección de bordes en una imagen y por esto es utilizado. Este método es trabajado por OpenCV lo cual hace mucho más fácil la forma de aplicación del algoritmo ya que este se encuentra como herramienta la cual es cvCanny().

Luego de haber hecho la detección de los contornos es necesario identificar vértices para saber que figura geométrica se encuentra en la imagen, así, si es posible encontrar un triángulo se deberían tener 3 vértices; si es un rectángulo 4 vértices y que entre ellos existan 90 grados y así aumentando el número de

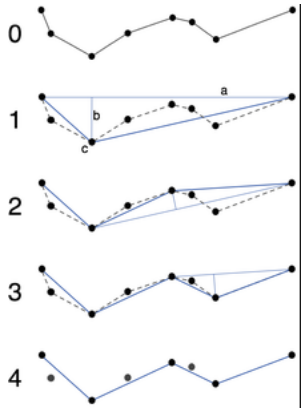
vértices si a su vez va aumentando la cantidad de lados. Si es un círculo identificado por una gran cantidad de vértices y una magnitud que denote el radio del mismo, existe un algoritmo que realiza la identificación de figuras geométricas y se denomina algoritmo de Douglas-Peucker.

ALGORITMO DOUGLAS-PEUCKER

Es de tenerse en cuenta que en el momento de tomarse un video las posibles figuras geométricas que en él se encuentran no son figuras geométricas perfectas; pueden que presenten curvaturas, esto hace que en el momento de generarse el contorno con el algoritmo Canny anteriormente mencionado se presenten curvas en estos contornos por lo que haría imposible identificar una figura geométrica, al haber una gran posibilidad de que no se generen las características definidas anteriormente para ser designada como una figura geométrica determinada, para solucionar este inconveniente fue creado este algoritmo.

El algoritmo de Douglas-Peucker tiene como objetivo reducir la cantidad de puntos que se encuentran en una curva y aproximarlos a una recta, lo que genera una disminución en la cantidad de puntos que componen la curva original, para esto se implementa un valor que nos sirve de tolerancia con el cual, se entraran a comparar todos los valores. Si cada punto se encuentra con un valor por encima a la tolerancia será modificado para linealizar mas los datos, en cambio si este punto está por debajo de la tolerancia este no será modificado.

Ilustración 6. Reducción de puntos en una curva por el método de Douglas-Peucker.



Fuente: Commons.

Con este método se garantizara encontrar de forma adecuada todos los vértices que se encuentren en el objeto que esté contenido en la imagen.

Anexo B. Función SMOOTH de MatLab.

Smooth response data

Syntax

yy = smooth(y)

gpuarrayYY = smooth(gpuarrayY)

yy = smooth(y,span)

yy = smooth(y,*method*)

yy = smooth(y,span,*method*)

yy = smooth(y,'sgolay',degree)

yy = smooth(y,span,'sgolay',degree)

yy = smooth(x,y,...)

Description

yy = smooth(y) smooths the data in the column vector y using a moving average filter. Results are returned in the column vector yy. The default span for the moving average is 5.

The first few elements of yy are given by

yy(1) = y(1)

yy(2) = (y(1) + y(2) + y(3))/3

yy(3) = (y(1) + y(2) + y(3) + y(4) + y(5))/5

yy(4) = (y(2) + y(3) + y(4) + y(5) + y(6))/5

...

Because of the way endpoints are handled, the result differs from the result returned by the filter function.

gpuarrayYY = smooth(gpuarrayY) performs the operation on a GPU. The input gpuarrayY is a gpuArray column vector. The output gpuarrayYY is a gpuArray column vector. This syntax requires the Parallel Computing Toolbox™.

Note: You can use gpuArray x and y inputs with the smooth function, but this is only recommended with the default 'method', 'moving'. Using GPU data with other methods does not offer any performance advantage.	
method	**Description**
'moving'	Moving average (default). A lowpass filter with filter coefficients equal to the reciprocal of the span.
'lowess'	Local regression using weighted linear least squares and a 1st degree polynomial model
'loess'	Local regression using weighted linear least squares and a 2nd degree polynomial model
'sgolay'	Savitzky-Golay filter. A generalized moving average with filter coefficients determined by an unweighted linear least-squares regression and a polynomial model of specified degree (default is 2). The method can accept nonuniform predictor data.
'rlowess'	A robust version of 'lowess' that assigns lower weight to outliers in the regression. The method assigns zero weight to data outside six mean absolute deviations.
'rloess'	A robust version of 'loess' that assigns lower weight to outliers in the regression. The method assigns zero weight to data outside six mean absolute deviations.

yy = smooth(y,span) sets the span of the moving average to span. span must be odd.

yy = smooth(y,*method*) smooths the data in y using the method *method* and the default span. Supported values for *method* are listed in the table below.

yy = smooth(y,span,*method*) sets the span of *method* to span. For the loess and lowess methods, span is a percentage of the total number of data points, less than or equal to 1. For the moving average and Savitzky-Golay methods, span must be odd (an even span is automatically reduced by 1).

yy = smooth(y,'sgolay',degree) uses the Savitzky-Golay method with polynomial degree specified by degree.

yy = smooth(y,span,'sgolay',degree) uses the number of data points specified by span in the Savitzky-Golay calculation. span must be odd and degree must be less than span.

yy = smooth(x,y,...) additionally specifies x data. If x is not provided, methods that require x data assume x = 1: length(y). You should specify x data when it is not uniformly spaced or sorted. If x is not uniform and you do not specify method, lowess is used. If the smoothing method requires x to be sorted, the sorting occurs automatically.

Anexo C. Coeficiente de correlación lineal de Pearson

Introducción

Antes de introducirnos en el modelo de regresión lineal, que hace referencia a la naturaleza de la relación entre distintas variables, pasaremos a exponer el estadístico utilizado para medir la magnitud de la relación (supuestamente lineal) entre dichas variables. Tiene sentido darle un tratamiento aparte por su importancia y las continuas referencias que ofreceremos a lo largo de este texto. Comenzaremos su desarrollo, por razones de simplicidad, para el caso particular de dos variables.

Coeficiente de correlación lineal de Pearson

El coeficiente de correlación de Pearson, pensado para variables cuantitativas (escala mínima de intervalo), es un índice que mide el grado de covariación entre distintas variables relacionadas linealmente. Adviértase que decimos "variables relacionadas linealmente". Esto significa que puede haber variables fuertemente relacionadas, pero no de forma lineal, en cuyo caso no proceder a aplicarse la correlación de Pearson. Por ejemplo, la relación entre la ansiedad y el rendimiento tiene forma de U invertida; igualmente, si relacionamos población y tiempo la relación será de forma exponencial. En estos casos (y en otros muchos) no es conveniente utilizar la correlación de Pearson. Insistimos en este punto, que parece olvidarse con cierta frecuencia.

El coeficiente de correlación de Pearson es un índice de fácil ejecución e, igualmente, de fácil interpretación. Digamos, en primera instancia, que sus valores absolutos oscilan entre 0 y 1. Esto es, si tenemos dos variables X e Y, y definimos

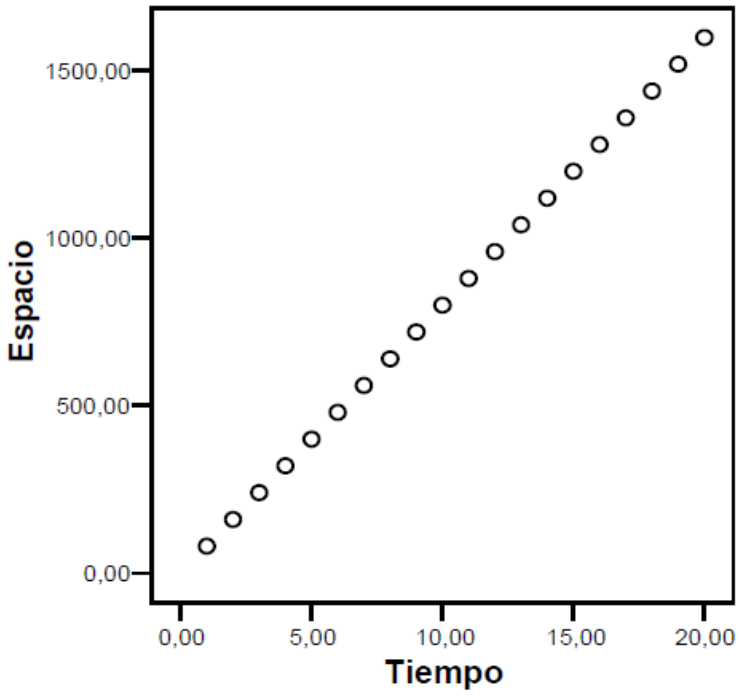
el coeficiente de correlación de Pearson entre estas dos variables como r_{xy} entonces:

$$0 \leq r_{xy} \leq 1$$

Hemos especificado los términos "valores absolutos" ya que en realidad si se contempla el signo el coeficiente de correlación de Pearson oscila entre –1 y +1. No obstante ha de indicarse que la magnitud de la relación viene especificada por el valor numérico del coeficiente, reflejando el signo la dirección de tal valor. En este sentido, tan fuerte es una relación de +1 como de -1. En el primer caso la relación es perfecta positiva y en el segundo perfecta negativa. Pasamos a continuación a desarrollar algo más estos conceptos.

Decimos que la correlación entre dos variables X e Y es perfecta positiva cuando exactamente en la medida que aumenta una de ellas aumenta la otra. Esto sucede cuando la relación entre ambas variables es funcionalmente exacta. Difícilmente ocurrirá en psicología, pero es frecuente en las ciencias físicas donde los fenómenos se ajustan a leyes conocidas, Por ejemplo, la relación entre espacio y tiempo para un móvil que se desplaza a velocidad constante. Gráficamente la relación ser del tipo:

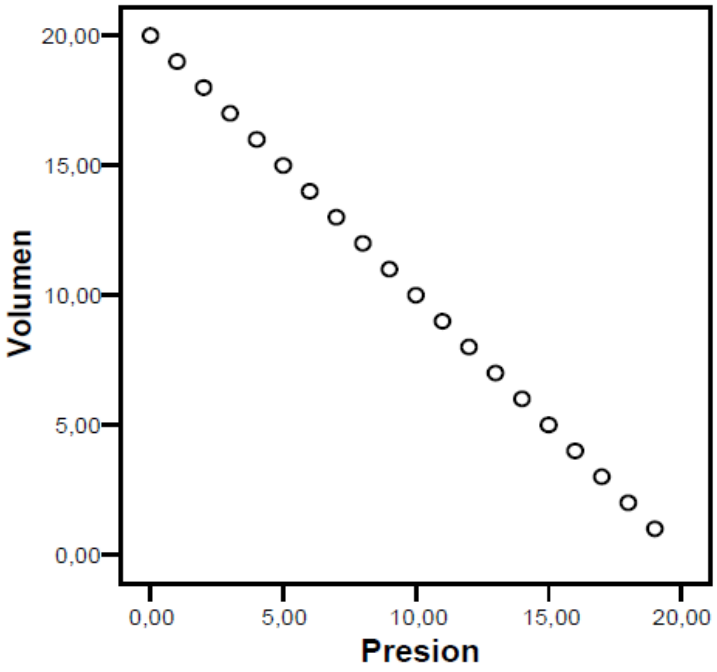
Ilustración 1. Correlación lineal perfecta positiva.



Fuente: Kovachi.

Se dice que la relación es perfecta negativa cuando exactamente en la medida que aumenta una variable disminuye la otra. Igual que en el caso anterior esto sucede para relaciones funcionales exactas, propio de las ciencias físicas. Por ejemplo, la relación entre presión y volumen se ajusta a este caso. El gráfico que muestra la relación sería del tipo:

Ilustración 2. Correlación lineal perfecta negativa.



Fuente: Kovachi.