UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE INGENIERÍA

CARRERA INGENIERÍA CIVIL

TEMA:

"Presentación y desarrollo de alternativa de medición de desplazamientos en ensayos de laboratorio empleando técnicas de procesamiento de imágenes para la Facultad de Ingeniería"

AUTORA:

Rojas Yela, Melissa Michelle

Trabajo de titulación previo a la obtención del título de

INGENIERO CIVIL

TUTOR:

Ing. Ponce Vásquez, Guillermo Alfonso M.Sc

Guayaquil, Ecuador

17 de septiembre del 2020

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE INGENIERÍA

CARRERA INGENIERÍA CIVIL

CERTIFICACIÓN

Certificamos que el presente trabajo de titulación, fue realizado en su totalidad por **Rojas Yela, Melissa Michelle** como requerimiento para la obtención del título de **Ingeniería Civil.**

TUTOR

f. _____
Ing. Ponce Vásquez, Guillermo Alfonso M.Sc

DIRECTORA DE LA CARRERA

f. _____
Ing. Alcívar Bastidas, Stefany Esther M.Sc.

Guayaquil, a los 17 del mes de septiembre del año 2020

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE INGENIERÍA

CARRERA INGENIERÍA CIVIL

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Rojas Yela, Melissa Michelle**

DECLARO QUE:

El Trabajo de Titulación, **Presentación y desarrollo de alternativa de medición de desplazamientos en ensayos de laboratorio empleando técnicas de procesamiento de imágenes para la Facultad de Ingeniería** previo a la obtención del título de **Ingeniería Civil**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance del Trabajo de Titulación referido.

Guayaquil, a los 17 del mes de septiembre del año 2020

LA AUTORA

f. _____

Rojas Yela, Melissa Michelle

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE INGENIERÍA

CARRERA INGENIERÍA CIVIL

AUTORIZACIÓN

Yo, **Rojas Yela, Melissa Michelle**

Autorizo a la Universidad Católica de Santiago de Guayaquil a la **publicación** en la biblioteca de la institución del Trabajo de Titulación, **Presentación y desarrollo de alternativa de medición de desplazamientos en ensayos de laboratorio empleando técnicas de procesamiento de imágenes para la Facultad de Ingeniería** cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y total autoría.

Guayaquil, a los 17 del mes de septiembre del año 2020

LA AUTORA:

f. _____

Rojas Yela, Melissa Michelle

URKUND

Urkund Analysis Result

Analysed Document:	Trabajo de titulo Melissa Rojas.pdf (D78844306)
Submitted:	9/9/2020 6:02:00 PM
Submitted By:	claglas@hotmail.com
Significance:	6 %

Sources included in the report:

https://www.infotechnology.com/online/Sistemas-operativos-de-codigo-abierto-cual-es-la-mejor-opcion-20191220-0008.html
https://www.instron.com.ar/es-ar/our-company/library/glossary/f/flexure-test
https://revistadigital.inesem.es/informatica-y-tics/opencv/
https://openwebinars.net/blog/los-5-mejores-editores-python/
https://www.famaf.unc.edu.ar/~pperez1/manuales/cim/cap2.html
http://www.scielo.org.mx/pdf/poli/n49/n49a8.pdf
https://omes-va.com/deteccion-de-colores/
https://omes-va.com/deteccion-de-colores2/

Instances where selected sources appear:

29

DEDICATORIA

A mis padres Sidonia Pilar Yela Alvarado y Edwin Remigio Rojas Llamasco, por ser el pilar en mi vida y darme el impulso de superación.

Melissa Michelle Rojas Yela

AGRADECIMIENTO

A Dios, por la gran bendición de tenerme con salud a mí y a mi familia para cumplir este proceso.

A mis padres, Sidonia Pilar Yela Alvarado y Edwin Remigio Rojas Llamasco por brindarme su apoyo incondicional, paciencia y amor que me ha permitido seguir siempre adelante con buenos valores y desempeño educativo.

A mis amigos y compañeros que uno forja durante la carrera. Gracias por su apoyo.

Al Ing. Guillermo Ponce, M.Sc. por las pautas necesarias durante el desarrollo de este trabajo.

Melissa Michelle Rojas Yela

UNIVERSIDAD CATÓLICA

DE SANTIAGO DE GUAYAQUIL

FACULTAD DE INGENIERÍA

CARRERA INGENIERÍA CIVIL

f. _____

ING. GUILLERMO PONCE VASQUEZ, M.Sc

TUTOR

TRIBUNAL DE SUSTENTACIÓN

f. _____

ING. STEFANY ALCÍVAR BASTIDAS, MSc.

DIRECTOR DE CARRERA

f. _____

ING. JAIME HERNÁNDEZ BARREDO, M.Sc

DOCENTE DE LA CARRERA

f. _____

ING. JOSÉ ANDRÉS BARROS CABEZAS, M.Sc.

OPONENTE

ÍNDICE

CONTENIDO DE ILUSTRACIONES

CONTENIDO DE CUADROS

CONTENIDO DE GRÁFICOS

RESUMEN

Este documento es producto de una investigación realizada bibliográficamente y en el campo de la experimentación, cuyo objetivo motivador para la realización de este proyecto es presentar, desarrollar e implementar una metodología práctica para realizar las mediciones de desplazamientos a partir del procesamiento de imágenes tomados de ensayos de laboratorio, de tal modo que la Facultad de Ingeniería tenga una nueva y versátil herramienta para optimizar las condiciones de enseñanza y mejorar el comportamiento de materiales y capacidad de los elementos. Para el desarrollo del proyecto se ha recurrido en la utilización del lenguaje de programación interpretador Python mediante su programa Editor Spyder que es con el que mejor se desarrolla, asimismo la utilización de la gran biblioteca OpenCv que funciona óptimamente con Python. El análisis y medición de los desplazamientos se hará con imágenes tomadas de videos tomadas de ensayos realizados experimentalmente de estructuras construidas a propósito para la ejecución del proyecto. Se ha tenido muy en cuenta obviamente la ventaja de la característica de ser programas de código abierto tanto de Python cuanto los editores y bibliotecas, se ha considerado puntualmente las aplicaciones técnicas para el análisis de las imágenes y sus colores, considerando también las escalas de grises y la disposición matricial y vectorial de los mismos con su resolución en pixeles, tamaño y combinación de colores.

Palabras clave: *mediciones, desplazamiento, intérpretes, editores, código abierto, software libre, programación, Python*

ABSTRACT

This document is the product of an investigation carried out bibliographically and in the field of experimentation, whose motivating objective for the realization of this project is to present, develop and implement a practical methodology to carry out displacement measurements from the processing of images taken from tests of laboratory, so that the Faculty of Engineering has a new and versatile tool to optimize the teaching conditions and improve the behavior of materials and capacity of the elements. For the development of the project, the use of the interpreting programming language Python has been used through its Editor Spyder program, which is the one with which it is best developed, as well as the use of the large OpenCv library that works optimally with Python. The analysis and measurement of the displacements will be done with images taken from videos taken from experimental tests of structures built on purpose for the execution of the project. Obviously, the advantage of being open source programs has been taken into account, both for Python as well as for editors and libraries, the technical applications for the analysis of images and their colors have been specifically considered, also considering gray scales. and their matrix and vector arrangement with their resolution in pixels, size and color combination.

Keywords: *measurements, displacement, interpreters, editors, open source, free software, programming, python.*

CAPÍTULO I
1 INTRODUCCIÓN

1.1 Generalidades

El uso de programación en la Ingeniería civil permite mejorar el procesamiento de imágenes, lo cual facilita la ejecución de las mediciones que se generan al realizar ensayos dinámicos o estáticos.

La computación científica es a menudo relacionada con la teoría, pero también tiene muchas características en común con el trabajo experimental. La principal ventaja es la facilidad de programación, minimizando el tiempo necesario para desarrollar, depurar y mantener el código. (Robert Johansson, 2016)

Frente a acciones dinámicas de alta intensidad, como algunos eventos que se han venido desarrollando como sísmicos, las estructuras de hormigón armado están diseñadas siempre para responder en el rango no lineal, lo que cambia significativamente la naturaleza de su comportamiento. La ampliación de las características discontinuas del hormigón por la fluencia de la armadura, así como la apertura y cierre de fisuras en la respuesta cíclica hace que los modelos analíticos y numéricos de la estructura se aparten evidentemente de la respuesta real que viene a desarrollarse en la estructura. Por ciertos ensayos que son experimentales en modelos físicos se hacen necesarias para una comprensión profunda del fenómeno estructural viene a darse en ellos. Los modelos de prueba a escala natural son económicamente inviables o materialmente imposibles debido al tamaño de las instalaciones reales.(Bertero & Bertero, 2014)

1.2 Objetivos

1.2.1 Objetivo general.

Presentar, desarrollar e implementar una metodología para mediciones de desplazamientos a partir de imágenes de ensayos de laboratorio, que permita a la Facultad de Ingeniería mejorar las condiciones de enseñanza de comportamiento de materiales y capacidad de elementos

1.2.2 Objetivos específicos.

- Presentar y desarrollar una metodología de medición de desplazamiento considerando las técnicas de procesamiento de imágenes y de fácil aplicación en los diferentes ensayos que se puedan realizar en la Facultad.
- Demostrar ejemplos de aplicación de las diferentes disposiciones de captura que se han empleado.

1.3 Alcance

Se generará un código que realizará el procesamiento de las imágenes de los ensayos, y entregará los desplazamientos que se hayan registrado.

Se llevará a cabo un manual beneficiario, para su fácil uso.

Se ejecutarán dos pruebas en ensayos estáticos.

Se ejecutarán dos pruebas en ensayos dinámicos.

1.4 Metodología

Para el desarrollo de la metodología se empleará el programa Python y se usarán videos capturados en dispositivos de uso común, como filmadoras y teléfonos celulares.

CAPÍTULO II.

2 REVISIÓN DE LITERATURA

2.1 Procesamiento mediante imágenes

2.1.1 Metrología visual.

Se conoce con esta denominación a la técnica para realizar mediciones de distancias con ayuda de imágenes; existen dos tipos de Metrología: de simple vista y de múltiples vistas; en la primera, para hacer una medición únicamente se requiere la captura de una imagen de una vista del objeto, es un método no invasivo fácil de utilizarse, se puede determinar varias distancias como una secuencia de imágenes y almacenar un registro histórico para análisis futuros. Esta técnica sirve más para mediciones de distancias sin contacto, con un emisor de ultra sonido, rayo láser o de luz infrarroja direccionado hacia un punto para medir el retorno de la señal emitida, mediante un receptor, éstos instrumentos digitales de medición de distancias incorporan chips de memoria para el registro histórico de mediciones, algunos pueden conectarse a una computadora por medio de puerto RS232, USB o Ethernet. En metrología de múltiples vistas, es necesario colocar varias cámaras en el espacio de manera determinista, y obtener valores de medición precisos según los atributos específicos de cada cámara; sin embargo, debido a que se analizan múltiples imágenes de la misma escena, La programación y el uso del sistema son más complicados .(Rodríguez et al., 2014)

La metrología es fundamental en la definición de bordes estableciendo las fronteras con lo cual se mejora la precisión en los procesos del tratamiento de las imágenes en procedimientos de control de calidad.(Gómez & Guerrero, 2016).

La obtención de una buena escena con cualquier tipo de cámara que se emplee depende de que se la ubique a la apropiada distancia al objeto, de tal manera que permita la focalización perfecta de los puntos que se van a analizar, lo que permitirá mayor precisión; por otra parte, el soporte de la cámara debe asentarse en un sitio donde no exista inestabilidad, por cuanto

el programa es lo suficientemente sensible para captar cualquier movimiento brusco y responder a eso con resultados que no sean realmente valiosos.

2.1.2 Procesamiento digital de imágenes

Debido a la complejidad de los procesos y la gran cantidad de cálculos que se deben realizar en el análisis y procesamiento de imágenes, se lo hace en computadoras, justamente porque la tecnología informática permite desde hace tres décadas la posibilidad de usar casi cotidianamente fórmulas matemáticas y otras herramientas de cálculo en estos procesos que exigen millones de operaciones casi instantáneamente con resultados de alta precisión, además se lo hace en equipos de uso doméstico como cámaras digitales y Smartphone; las primeras áreas en utilizar la digitalización de imágenes fue la medicina, en radiología. Además de la aparición de equipos tecnológicos y de software, el uso de conceptos de física y matemáticas como métrica y entropía en algoritmos para procesos de diseño ha ayudado a la optimación de la manipulación digital de imágenes en tiempo real, hoy se integran formatos de lectura y representación de imágenes, operaciones de modificación, transformación de tonalidades en colores, y generación de efectos sobre regiones de una imagen.(Pérez & Valente, 2018)

Al digitalizarse un código en el programa, en el momento que es captado brinda oportunidades para la aplicación de técnicas analíticas que permitan desarrollar imágenes lógicas para identificar objetos, operaciones y recursos de orientación.

El gran análisis automático que se da al procesar una imagen cuando es leída, facilita la ejecución de procesos desarrollados lo más rápido posible, con resultados que exige el cumplimiento de un objetivo, de tal manera que, el utilizar este método ha generado que el programa sea de gran ayuda; del mismo modo, los últimos avances en tecnología de reconocimiento de imágenes toman gran importancia para la aplicación que se le está dando en la Ingeniería civil.

2.1.3 Formato de imagen y representación digital

Las imágenes son un lenguaje usado para transmitir mensajes, símbolos y otros tipos de información, contando con un soporte para su representación digital para después poder modificar su contenido visual y simbólico; una imagen digital es aquella desarrollada mediante herramientas de computación, se diferencia de la fotografía, que refleja la imagen de un objeto real, y de la pintura, que es netamente producto de la creación o de la imaginación de un artista. El procesamiento o transformación digital de imágenes con aplicación de programas informáticos o sistemas Inteligentes que emulan las características y actitudes de la inteligencia humana, las imágenes digitalizadas se almacenan en dispositivos de disco o memoria bajo formatos definidos para aceptar diferenciación de color en histogramas de imagen con el número de pixeles que corresponden a cada tono.(Gómez & Guerrero, 2016)

El algoritmo para la digitalización de una imagen contiene los pasos de captura de la imagen mediante cámara digital u otro dispositivo, para realzar y mejorar la apariencia visual de la imagen o para restaurar imágenes degradadas, luego la segmentación y umbralización que es la división de la imagen en áreas significativas definiendo sus bordes, la extracción de características, para lograr las definiciones de la imagen esperadas.(Gómez & Guerrero, 2016)

Una imagen en escala de grises es una matriz representada por 1 píxel que equivale 1 byte, acepta 256 niveles de grises que van desde el 0, que es color negro, hasta 255, que es color blanco; la matriz de una imagen en color se representa con 1 píxel que es igual a 3 bytes, donde cada píxel tiene los valores: (Rojo, Verde, Azul).(Rodríguez et al., 2014)

La mayor resolución que se tenga en una imagen, significa que tenemos mayor cantidad de puntos (píxeles) en toda la imagen, en cambio, si la imagen tiene una resolución menor, es decir, menor cantidad de pixeles, será motivo para que haya menos precisión al momento de intentar detectar pixeles para

el desarrollo de conclusión, la resolución de la imagen es directamente proporcional a la precisión de las coordenadas.

2.1.4 Bandas en imágenes digitales

Sólo es posible obtener una imagen remota si se cuenta con algún tipo de interacción entre el objeto observado y el detector, para imágenes digitales dependiendo del tipo de radiación electromagnética que puede detectar, asimismo, la información extraída de un objeto depende de la interacción entre la radiación y el objeto, esto da origen al concepto de bandas, que divide el espectro electromagnético en función de los tipos de interacción entre radiación y materia, con lo que se definen desde los objetos a analizar hasta los detectores y materiales que pueden utilizarse, lo cual se cumple porque los objetos a partir de sus propiedades físicas dentro del espectro electromagnético dependiendo de su longitud de onda, intensidad y tipo de radiación absorben, reflejan o emiten cuantos de energía, los rayos X, la radiación ultravioleta, infrarroja o de microondas, pueden recabar esta información con uso de detectores de forma digital o analógica. (Pérez & Valente, 2018)

La información extraída de un objeto depende de la interacción entre el objeto observado y el detector de imágenes digitales según el tipo de radiación electromagnética que puede detectar; de esta relación entre objeto y radiación se origina el concepto de bandas, que divide el espectro electromagnético en función del tipo de interacción entre radiación y materia.

2.1.5 Representación digital: mapa de bits (bitmaps)

Un Bitmap es la representación elemental de imágenes digitales para ser cargada en la memoria de un computador, se conforma con arreglos vectoriales o matrices, dispuestos ordenadamente, en el caso de imágenes bidimensionales o 2D, se ordena los *pixeles o* elementos de la matriz en filas, asignando a cada punto un valor que determina el color en esa posición de la imagen. Para imágenes en tonalidades grises, el valor de cada elemento de la matriz es un escalar, en cuanto a imágenes en color, el valor de cada

elemento de matriz es un vector de tres coordenadas, donde se especifica el grado de influencia de los colores rojo, verde y azul: Red, Green, Blue, es decir RGB; esta codificación también puede ser CMYK, de Cián, Magenta, Yellow y Black.

Usualmente se emplean escalas en rangos dinámicos o 2^N*, donde N es la cantidad de* bits, en el caso más común de 8-bits, la escala se define en el rango [0, (2^N−1)], o sea [0,255], (N = 8). El uso típico de 8-bits es por dos motivos: según estudios biométricos el ojo humano no es tan sensible como para diferenciar más de 256 niveles de intensidad en un color, y por la capacidad de almacenamiento en el computador que está en el rango de 2^8. Los colores en el rango visible pueden representarse como combinaciones RGB, variando desde el negro (0, 0, 0) al blanco (255, 255, 255). Por lo tanto, una imagen RGB.(Pérez & Valente, 2018)

Las imágenes digitalizadas son el formato ideal que se utiliza para cargarlas en la memoria de un computador, tienen la forma de arreglos vectoriales o matrices, dispuestos ordenadamente para representar imágenes bidimensionales o en 2D, donde los pixeles que son los elementos de la matriz van dispuestos en filas, el valor que toma cada punto determina el color de la imagen en esa posición; por ejemplo, en las imágenes en tonalidades grises, el valor de cada elemento de la matriz es un escalar, para las imágenes en color, el valor de cada elemento de matriz es un vector de tres dimensiones o coordenadas que especifican el grado de influencia de los colores rojo, verde y azul

2.1.6 Representación vectorial de imágenes y modificación de colores

Toda imagen está definida matemáticamente o vectorialmente por medio de ecuaciones que describen perfectamente los contornos y rellenos de cada ilustración, por eso se puede aplicar *scaling* sin riesgo de pérdida de calidad y sin variabilidad en la formación o reproducción en dispositivos, esto toma mayor relevancia en casos de ilustraciones con marcadas zonas con contornos curvados, donde a menor presencia de pixeles se perdería

resolución. Se puede representar la diferencia entre dos colores con representación digital vectorial RGB calculando la distancia entre los valores que los representan, por ejemplo, si tenemos un color cuyo vector es $C_1 = (R_1, G_1, B_1)$ y otro color con vector $C_2 = (R_2, G_2, B_2)$, la distancia entre éstos dos colores estará dada por el vector resultante $D\ (C_1, C_2) = (R_1 - R_2)_2 + (G1 - G2)_2 + (B_1 - B_2)_2$, es una suma vectorial cuya resultante es un nuevo color con tonalidad diferente a los originales; análogamente se procede para el caso de colores en tonalidades en grises.(Pérez & Valente, 2018)

Se puede aplicar scaling sin perder calidad y sin variabilidad en la formación o reproducción de imágenes en dispositivos visuales, toma mayor relevancia en ilustraciones con contornos curvados, porque a menor presencia de pixeles menor resolución; asimismo la diferencia entre dos colores RGB se definen por la distancia entre los valores que los representan, siendo que con OpenCV se realiza con HSV.

2.1.7 Resolución, tamaño de imagen y tamaño de archivo

Comúnmente la resolución de una imagen se define como la calidad o fidelidad que tenga, o sea, a mayor resolución, mayor calidad o nitidez de la imagen; esto puede entenderse en términos de pixeles que mientras mayor cantidad de pixeles componen la imagen, más nítida, de mejor calidad, o de mejor definición es; en conclusión, la resolución de una imagen es la capacidad para percibir los detalles de la imagen dada por la cantidad de pixeles por pulgada cuadrada que contiene, de ahí tenemos la medida ppi = pixeles por inch. Dicho en otras palabras, la calidad de la imagen se representa depende de la resolución, o sea, de la cantidad de memoria existente para generar el gráfico.

Se entiende como tamaño de imagen a las medidas de alto y anchura que la imagen impresa ocupa en un plano, en cambio, se define al tamaño de archivo como la cantidad de memoria física que se necesita para almacenar la información de la imagen digitalizada en algún medio de almacenamiento informático. La resolución por su parte es la cantidad de pixeles que la imagen ocupa en cada pulgada cuadrada; de esto se generan varias conclusiones,

por ejemplo, si en un dispositivo digital se aumenta el tamaño de la imagen, la calidad de esta disminuye porque la resolución existente en el equipo es fija, se dispondrán menos pixeles por pulgada cuadrada de imagen. (Pérez & Valente, 2018)

Aproximadamente los detalles de los videos que se han ido procesando sin dificultad alguna ha sido:

Cuadro N° 1 Detalle de propiedades de video

Video		***Archivo***	
Ancho fotograma	1024	***Tipo de elemento***	MP4
Alto fotograma	576	***Tamaño***	153 MB
Velocidad fotograma	50.00 fotogramas/seg		

Elaborado por: Melissa Rojas Yela, obtenido por propiedades de video

Para el ingreso de videos en el programa lo que se ha realizado es que la velocidad fotograma sea de 50.00 fotogramas/ segundo dando así que la lectura que el programa de sea esa cantidad de frame, es un paso que se debe hacer porque de no ser así se debe ajustar los códigos para que sea de acorde al detalle que se tenga en el video a procesar.

Contraste en una imagen

El contraste de una imagen se refiere básicamente a la incidencia visual de la cantidad de grises en la resolución de la misma, técnicamente esto se entiende como el aumento o disminución de la pendiente de la recta que define la imagen, que normalmente es de 45 grados para una imagen cuyo rango es de (0, 255).

2.2 Lenguaje de programación Python

Es un lenguaje de programación de alto nivel y código abierto que actúa como un programa interpretador, puede ser aplicado en distintas plataformas con

diferentes paradigmas, es decir, puede analizar y ejecutar otros programas, incluidos aquellos orientados a objetos, de programación imperativa y programación funcional, en diferentes sistemas operativos como Windows, Linux o Mac; sus estructuras de datos organizados en diccionarios, listas, conjuntos y tuplas, le permite realizar tareas complejas con carácter minimalista en pocas líneas de código de fácil legibilidad. Python es administrado por la fundación sin fines de lucro *Python Software Fundation* License, con licencia de código abierto, fue creado a finales de los años 80 por el programador holandés Guido van Rossum con base en el lenguaje ABC con influencia de Lenguaje C, Algol 60, Modula-3 e Icon, hoy es parte del navegador Google. (Challenger Pérez et al., 2014, pp. 2–3)

Después de su versión original, Python se ha desarrollado en versiones 2.0, 2.6 y otras, hasta la versión actual 3.0 superior a las anteriores, en la actualidad Python se aplica en los campos de inteligencia artificial y machine learning pero siempre buscando tener metodologías más sencillas para escribir códigos, los desarrolladores de Python mantienen algunos principios de diseño para escribir sus códigos; por ejemplo, bello es mejor que feo, explícito es mejor que implícito, simple es mejor que complejo, complejo es mejor que complicado, plano es mejor que anidado, disperso es mejor que denso, lo práctico vence a lo formal, los errores no deben pasar desapercibidos, descarte la tentación a adivinar, debe haber una y preferentemente una sola manera obvia de lograr algo, mientras más sencilla y clara sea una idea será mejor, ahora es mejor que nunca; etc. (Challenger Pérez et al., 2014, pp. 3–4)

Existen diversas razones para utilizar el lenguaje de programación Python en la representación de las imágenes digitalizadas, por una parte es un lenguaje de código abierto con lo cual los usuarios o programadores tienen la posibilidad de manipular en el código fuente, es de software libre con lo cual se evita el gasto económico, por otra parte, es multi paradigmático y puede ser abierto en distintas plataformas, se lo puede desarrollar para otros programas visuales y orientados a objetos; en definitiva es muy versátil y fácil de usarse.

El formato de imágenes que se mantiene en pyhton tiene como:

Cuadro N° 2 Soporte de formato de imagen Soporte de formato de imagen

FORMATO	*LEE*	*ESCRIBE*	*DESDE LA VERSIÓN*	*NOTA*
PNG	X	X	0.1	
PNG (16bits)	X	X	0.4	
TIFF	X	X	0.1	
JPEG	X	X	0.1	
WEBP	X	-	0.2	
BMP	X	-	0.2.5	Solo se admiten mapas de bits sin comprimir
STK	X	-	0.2	
LSM	X	-	0.2.2	
XCF	X	-	0.2.2	Solo si se usa “xcf2png” encontrándose en la ruta

Obtenido de: (Coelho, 2013)

2.2.1 Modo interactivo, elementos del lenguaje, indentación y variables.

A diferencia de los lenguajes de programas compiladores, el modo interactivo del intérprete de Python estándar permite ingresar las instrucciones una por una, como un intérprete de comandos, de tal modo que pueden ser evaluadas inmediatamente en porciones de código antes de ser integradas como parte del programa, lo cual facilita la labor de los desarrolladores del software principiantes y avanzados. A diferencia de otros lenguajes, Python utiliza palabras en vez de símbolos, lo que permite que sean leídos con facilidad,

así, los operadores lógicos: !, || y &&, Python los escribe como: not, or y and. Un intérprete lee un programa escrito en lenguaje de alto nivel instrucción por instrucción, traduce y ejecuta los códigos de máquina simultáneamente.(Marzal et al., 2014)

Los bloques de código, sean estos bucles, funciones o clases, se delimitan con espacios o tabuladores ubicados antes de cada línea de órdenes de cada bloque, a esto se conoce como Indentación; otros lenguajes de programación prefieren declarar los bloques de códigos con un conjunto de caracteres escritos entre llaves o corchetes; la Indentación permite escribir cada instrucción en una sola línea, pero por razones de legibilidad se puede dividir la instrucción en varias líneas, separándolas con una barra invertida \ al final de una línea, y se entiende que la instrucción continúa en la línea siguiente. Las *variables* pueden definirse en forma dinámica sin especificar de antemano cuál es su tipo, incluso puede tomar distintos valores de diferente tipo, en diferentes momentos. Se usa el símbolo = para asignar valores, por ejemplo, puede ser x = 1 en un momento y x = "texto" en otro. Los nombres de variables pueden contener números y letras pero deben comenzar por una letra, además existen 28 palabras reservadas.(Marzal et al., 2014)

El lenguaje de programación Python no es un compilador, es un programa interpretador que ejecuta las instrucciones línea a línea, lo cual lo hace mucho más ágil que los compiladores.

2.3 El editor Spyder (*Scientific Python Development Environment*)

Se entiende que los cinco mejores editores para el lenguaje de programación Python son: PyDev para Eclipse, PyCharm, VIM, Wing, Spyder Python. De estos, Spyder es el IDE de código abierto más adecuado para el desarrollo científico de Python, por ser un ágil y potente editor de software libre, escrito en Python bajo licencia del MIT. Spyder Python actúa como un editor de varios lenguajes, su consola interactiva, el visor de documentación, el explorador de variables, el explorador de archivos, etc. Spyder IDE tiene versiones para Windows, Mac o Linux. Además de todo ello, puede ser utilizado como una

biblioteca de extensión PyQt y puede ser incorporado en aplicaciones que proporciona widgets potentes relacionados con la consola para nuestras aplicaciones basadas en PyQt. Se puede utilizar para integrar una consola de depuración directamente en el diseño de su interfaz gráfica de usuario (Ortego, 2017)

Spyder es un entorno científico escrito para Python, diseñado para científicos, ingenieros y analistas, su combinación de funciones avanzadas: edición, análisis, depuración, creación de perfiles, introspección y un entorno informático numérico, lo convierte en una potente herramienta de desarrollo integral, con la exploración de datos, ejecución interactiva, inspección profunda con capacidades de visualización de paquete científico, capacidades que se pueden ampliar, más su sistema de complementos y aplicaciones; asimismo Spyder puede ser una biblioteca de extensión de PyQt5, con esto los desarrolladores utilizan su funcionalidad para integrar sus componentes, como la consola interactiva, en su propio software PyQt. (González, 2018)

Spyder trabaja de manera eficiente como editor en varios idiomas con un navegador de funciones o clases, herramientas de análisis de código, finalización automática de código, división horizontal o vertical y definición de acceso. Las consolas IPython trabajan con la flexibilidad de una interfaz GUI completa; ejecute su código por línea, celda o archivo y renderiza gráficos en línea. Es un Explorador dinámico de variables, puede Interactuar y modificar las variables sobre la marcha; como perfilador elimina cuellos de botella para desencadenar el rendimiento de su código, como depurador para hacer el seguimiento paso a paso de la ejecución de su código de forma interactiva. (Ortego, 2017)

Al igual que el lenguaje de programación Python, el editor Spyder es de código abierto y de software libre, razones que lo hacen de más fácil uso, además, es el editor que mejor se adapta al lenguaje Python.

2.3.1 Sistemas de código abierto.

Es una forma de desarrollo de software creada para otorgar beneficios prácticos a los usuarios, como tener acceso al código fuente, su ventaja no es

solamente por su uso gratuito como los de software libre. La mayoría de empresas públicas y de gobiernos del mundo utilizan plataformas de código abierto como LINUX para la gestión de sus sistemas de información, porque son sistemas seguros y económicos, son varias las ventajas de estos sistemas, por ejemplo, facilita la toma de decisiones según lo que ocurre en la comunidad de código abierto que lo desarrolla, tanto en el soporte cuanto en la velocidad y estabilidad de actualizaciones, obteniendo soluciones innovadoras, robustas, más seguras y notorias, aumentando su rendimiento, con entornos de usuario amigables. Las proveedoras de software de código abierto brindan mayor apoyo y más oportuno a sus clientes porque el código de este tipo de software está disponible para todos, con gran capacidad de soporte para el cliente. (Infotecnology, 2019)

Según dice Adrián Cambareri “para poder convertir un ecosistema en expansión en un entorno realmente híbrido, se necesita un sistema operativo que cumpla los siguientes requisitos: escalabilidad, transferencia de las cargas de trabajo sin complicaciones y gestión de aplicaciones que funcionan en todas partes” (*Adrián Cambareri, Gerente de la Unidad de Negocios de Red Hat Enterprise Linux para Red Hat Latinoamérica)*. (Infotecnology, 2019)

Un sistema de código abierto no implica la gratuidad de su uso sino la posibilidad de interpretar los códigos línea a línea desde el código fuente, lo cual permite corregir errores sobre la marcha mientras se desarrollan los procesos.

2.3.2 Programa interpretador.

Estos programas realizan la traducción de las instrucciones paso a paso al mismo tiempo que ejecuta ordenadamente los procesos, instrucción por instrucción, no guardan el resultado de la traducción que realizan; usando un intérprete, un solo archivo fuente puede producir resultados iguales incluso en sistemas sumamente diferentes, los programas interpretadores suelen ser más lentos que los compiladores y los ensambladores debido a la necesidad de traducir el programa mientras se ejecuta, pero son más flexibles como entornos de programación y depuración, en la actualidad, uno de los entornos

más comunes de uso de los intérpretes es en los navegadores web debido a la posibilidad que estos tienen de ejecutarse independientemente de la plataforma que lo contiene. (Marzal et al., 2014)

Los intérpretes son diferentes a los programas compiladores, mientras estos últimos primero leen el código como un todo, los intérpretes leen el programa fuente línea por línea y ejecutan cada línea directamente en la plataforma sin antes traducir el código; es decir, cada análisis se realiza durante la ejecución del programa. Los intérpretes ejecutan las instrucciones del programa sin antes traducir el código fuente; es decir, el intérprete implementa las rutinas registradas, exactamente como la puso el desarrollador.(Marzal et al., 2014)

Cuando se interpreta el código fuente, la entrada y el código fuente se ejecutan al mismo tiempo, entonces, el intérprete lee cada línea con una instrucción en forma inmediata, como si desarrollase los dos pasos simultáneamente, es decir, la traducción y la ejecución se cumplen en orden lógico subsiguientemente, según las especificaciones del código fuente. Los compiladores son programas muy eficientes sobre todo para la depuración, son precisos para identificar errores, inmediatamente dejan de funcionar cuando detectan errores, el sistema se para; en cambio, si el intérprete se detiene, los programadores inmediatamente resuelven el error y el sistema sigue. Como desventaja de los intérpretes tenemos que son bastante lentos comparados con los compiladores, en cada línea lee inclusive elementos repetitivos, sin embargo, este problema de falta de velocidad se resuelve con modificaciones como el JIT (Just-in-time compiler) o el intérprete de bytecode.(Marzal et al., 2014)

Los programas intérpretes son más agiles que los programas compiladores, porque sin necesidad de leer el código completo leen desde el programa fuente y ejecutan cada línea de códigos directamente sin traducir el código. Los intérpretes ejecutan las instrucciones del programa mientras lee cada línea.

2.4 OpenCV

Es una biblioteca de visión artificial, es de uso libre y se la entiende como una visión general abierta para la detección de movimiento, reconocimiento de objetos y reconstrucción 3D a partir de imágenes, está en el mercado desde finales del siglo XX y fue creada por Intel, es de los de Software libre; entre sus principales características tenemos que es compatible con diferentes Sistemas Operativos, entre ellos: GNU/Linux, Mac OS X, Windows y Androide, funciona en diversas arquitecturas de hardware x86 y x64 para PC, y ARM para Smartphone. OpenCV contiene más de 2500 algoritmos**,** su biblioteca tiene interfaces para múltiples lenguajes, incluidos Python, Java y C++.

Para usar bien esta librería, se necesita tener conocimientos en Librerías Numpy y Matplotlib; para módulos principales se ejecuta *pip install opencv-python,* para módulos principales y adicionales (contrib) se debe ejecutar **pip** install opencv-contrib-python, estos comandos se utilizan con Jupyter o cualquier IDE de Python.(Marín, 2020)

Cuenta con una activa documentación de referencia completa y actualizada para desarrolladores a quienes franquea el código fuente, con ejemplos y tutoriales sobre sus funciones, es accesible a usuarios inexpertos en visión artificial, su entorno es de fácil uso y desarrollo; su programación está desarrollada en lenguaje C y C++ orientado a objetos, con las capacidades de los procesadores multinúcleo. OpenCv se aplica para caracterizar en 2D y 3D para reconocimiento facial, reconocimiento de colores, adaptación de cámara, Reconocimiento de gestos, Interacción entre usuario y computadora, Robótica móvil, reconocimiento de objetos, comprensión de movimientos, segmentación, estereoscopía; etc. OpenCV está totalmente desarrollado en C++, orientado a objetos y con alta eficiencia computacional. Su API es C++ con conectores para otros lenguajes como: Python, Java, Matlab, Octave, Javascript; su documentación y tutoriales se priorizan según los lenguajes, C++, Python, Java, Javascript. (Marín, 2020)

2.5 Imágenes como matrices

Toda imagen es una matriz estándar de Numpy, con puntos de datos llamados pixeles, mientras mayor sea el número de píxeles la imagen tendrá mejor resolución, cada pixel es como un pequeño bloque de información en dos dimensiones, según la profundidad del píxel es la información sobre el color. Las imágenes para ser procesadas en la computadora, deben convertirse una forma binaria. Se puede calcular el color de cada imagen elevando el número de color exponencialmente al número de bits por pixel; entonces, mientras mayor sea la cantidad de bits/píxel, habrá más colores posibles en las imágenes.

El siguiente cuadro ilustra esta relación:

Cuadro N° 3 Relación de posibles colores en las imágenes.

Bit/Pixel	Possible colours
1	2^1=2
2	2^2=4
3	2^3=8
4	2^4=16
8	2^8=256
16	2^16=65000

Fuente: (Marín, 2020)

Según esto se determina la representación de los diferentes tipos de imágenes: por ejemplo, Una imagen binaria consta de 1 bit/píxel, por lo tanto, solo puede tener dos colores posibles, es decir, blanco o negro, donde negro está representado por el valor cero, el blanco se representa con el número 1. Las imágenes en escala de grises constan de 8 bits por píxel, entonces, puede tener 256 sombras diferentes $2^8 = 256$. En cuanto a las imágenes en colores `primariamente pueden ser entre rojo, azul y verde, la gama de colores resulta de la combinación de estos tres colores primarios, en las proporciones requeridas. (Marín, 2020)

Se entiende entonces que cada imagen en color consta de 8 bits por píxel por lo que se pueden representar en 256 tonos de colores diferentes. Las imágenes pueden ser representadas como u arreglo o matriz de tres dimensiones.

La siguiente imagen ayudará a realizar los rangos de los colores para obtener detección de diferentes colores que se desee.

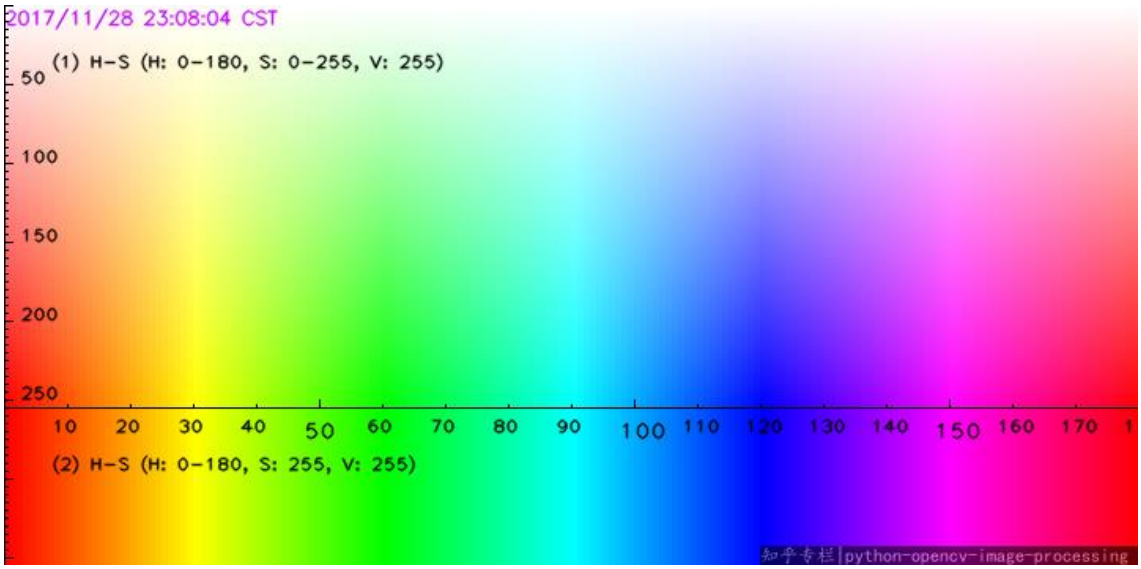


Ilustración 1 Vista de los componentes HSV
Imagen obtenida de una respuesta en stackoverflow por (Solano, 2019a)

OpenCV permite realizar las siguientes operaciones simples con imágenes**:**

- Abrir y guardar imágenes
- Dibujar formas simples en imágenes
- Escribir en imágenes

Son las operaciones básicas para crear una base antes de avanzar y poder utilizar todas funciones más complejas en OpenCV.

También se puede importar imágenes en OpenCV mediante los siguientes pasos:

1. Importar las bibliotecas necesarias
2. Leer la imagen con la función *imread*.

3. Elegir el tipo y la forma de la matriz.

Las imágenes creadas o importadas pueden guardarse en el directorio de trabajo. Otras funciones de OpenCV son: dibujar imágenes, dibujar en imágenes, todo esto se hace ingresando las respectivas coordenadas para longitud, anchura y profundidad, asimismo se puede agregar texto a las imágenes. También se puede realizar análisis y tratamiento de imágenes desde detectar caras y clasificarlas según género hasta crear modelos de realidad aumentada o usar clasificadores para detectar objetos. (Marín, 2020)

Formato que puede llevar el video para ser procesado.

FourCC es un código de 4 bytes que se utiliza para especificar el códec de vídeo. La lista de códigos disponibles se puede encontrar en fourcc.org. Depende de la plataforma.

Códecs:

- En Fedora: DIVX, XVID, MJPG, X264, WMV1, WMV2. (XVID es más preferible. MJPG resulta en vídeo de gran tamaño. X264 da vídeo de tamaño muy pequeño)
- En Windows: DIVX (Más para ser probado y añadido)
- En OSX: MJPG (.mp4), DIVX (.avi), X264 (.mkv).

2.5.1 Técnicas de marca por color.

Siendo importante el desarrollo de este proceso, para que exista una buena captación al momento de ser grabado y que el programa pueda bien procesar las imágenes, por lo siguiente se toma en cuenta:

- Donde se procede a añadir el color que va a ser captado
- Color ideal que al programa se le pueda dar con los rangos que se puede sacar en la ilustración 1, una vez analizado el tipo de color debe entrar en ese rango.
- Tener presente las marcas que se creen de una forma limpia, clara y precisa

Sea cual sea el material que se usa para marcar el color se debe ser claro y mantener un color en sí que el programa detecte para la precisión de coordenadas que se efectúa al proceder la ejecución del video de un tipo de ensayo.

2.6 Ensayo a flexión

Es un método empleado para medir el comportamiento de materiales sometido a una carga. Al analizar la viga en un ensayo de flexión, siendo una prueba estática, que puede determinar el módulo de flexión, el esfuerzo de flexión y la deflexión del material.

Las siguientes normas para realizar la resistencia de un material:

- ASTM D-790 en plásticos;
- ASTM C-674 en cerámica blanca cocida;
- ASTM D-797 en elastómeros;
- ASTM A-438 en hierro fundido; y,
- ASTM D-86 en vidrio (Instron. (SA), 2020)

2.7 Ensayo a corte

Cuando existen grandes cargas concentradas las áreas cercas son afectadas, y las cercas de los apoyos.

Siendo directo el corte en un área, este ensayo se puede realizar con una probeta empotrada, colocando una carga y obteniendo esfuerzos de flexión, corte, y tensión en dicha área. Para obtener esfuerzo de corte máximo aplicando la carga máxima hasta fracturar.

Ilustración 2 Elemento empotrado de un lado

Elaborado por Melissa Rojas Yela

También se tiene al caso en el que se empotra los dos extremos colocando una carga que este va a obtener dos áreas de corte.

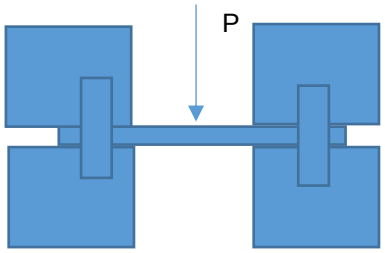


Ilustración 3 Elemento empotrado en los dos extremos

Elaborado por: Melissa Rojas Yela

Y el caso similar al ensayo de Flexión para una viga, teniendo grietas por cortante que pueden presentarse en:

$$1 < \frac{a}{d} \leq 2{,}5$$

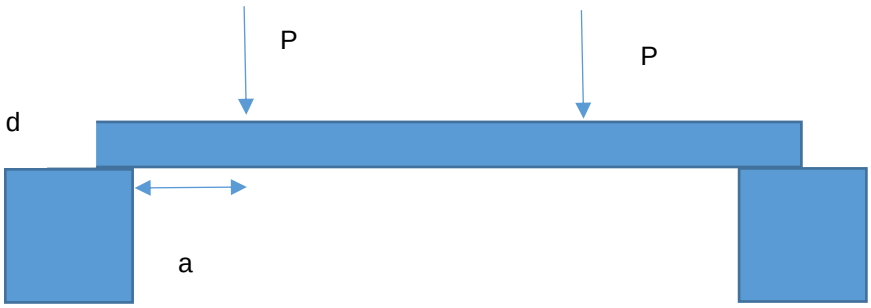


Ilustración 4 Viga que puede fallar por cortante

Elaborado por: Melissa Rojas Yela

2.8 Ensayo dinámico

Los efectos sísmicos de gran magnitud, hacen necesario que se realice este tipo de ensayos en los que se analizan una estructura a pequeña escala en una mesa vibratoria, este tipo es estudios siguen siendo la mejor opción para comprender de excelente manera el comportamiento dinámico para estructuras civiles y para poder visualizar los efectos reales que tienen dicho comportamiento en la vida real.

Magnitudes de interés en propiedades dinámicas

- Aceleraciones
- Desplazamientos absolutos
- Fuerzas elásticas
- Momentos en la base y cabezal
- Ductilidad de desplazamientos

Los modos de vibración son debido a la definición de una matriz de masas (“m”) y de rigidez (“k”), en base a estas matrices se obtiene sus correspondientes de valores propios, permitiendo conocer la frecuencia natural de dicho modo, a partir de la frecuencia natural se obtiene el período natural. De los valores propios se pueden conocer sus vectores propios de cada grado de libertad dentro de ese modo, a través de esos vectores propios se conoce la forma modal de la estructura.

Para estos ensayos se tienen periodos es el tiempo que tarda una estructura en ir y regresar a un mismo punto significando que es completar un ciclo sin importar que tan lejos vaya que esto puede ocasionar la mesa vibratoria obteniendo periodos largos o cortos dependiendo del modelo que se emplee.

CAPÍTULO III

3 RESPUESTAS DE DESPLAZAMIENTOS

El programa muestra los resultados en Excel, a los cuales se llega mediante el producto de dos factores, los pixeles del video por la distancia entre los puntos en el ensayo.

3.1 Ensayos estáticos

3.1.1 Ensayo 1: Viga de madera balsa a flexión.

Se usó un soporte para colocar el elemento de madera de balsa de forma horizontal, de tal manera que se encuentre simplemente apoyado en ambos extremos. Se procedió a usar el peso a colocar en un punto central de la viga. Cabe destacar que se realizó este proceso con una longitud efectiva del elemento de 53 cm y se colocó un punto a 2cm como referencia para el proceso en el programa.

A continuación, se muestra el ensayo a flexión

Ilustración 5 Elemento simplemente apoyado con una carga Puntual en el centro del claro. Obtenida de: Melissa Rojas Yela

3.1.2 Propiedades del video.

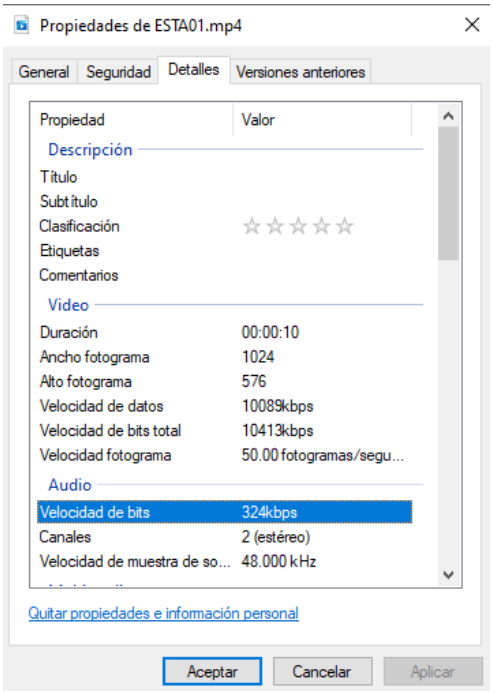


Ilustración 6 Propiedades de video del ensayo a flexión.

Capturado por: Melissa Rojas Yela

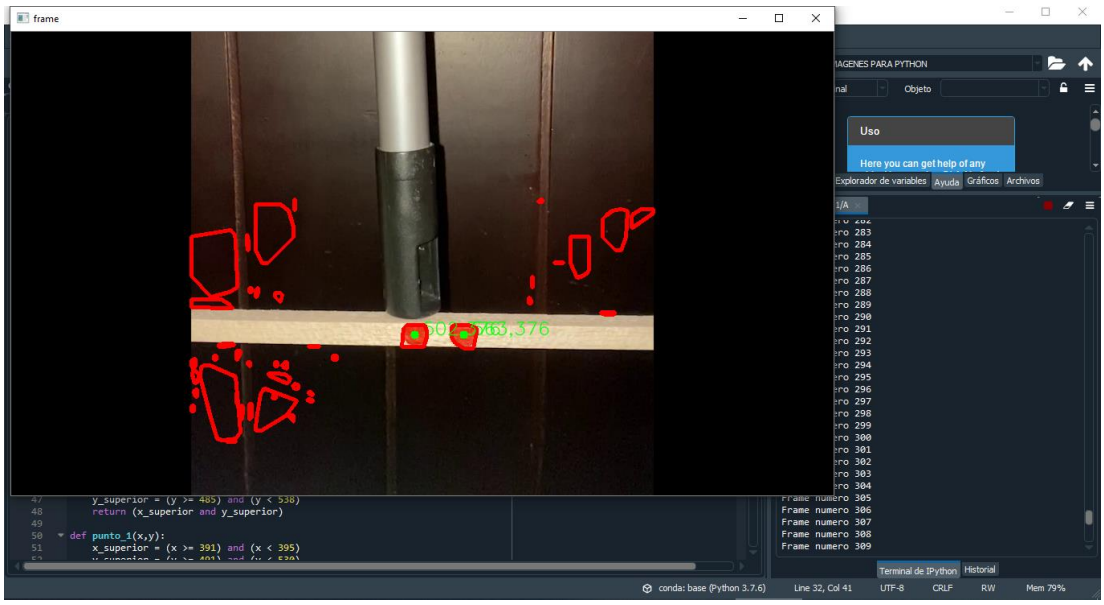


Ilustración 7 Lectura de coordenadas de pixeles en el programa

Capturado por: Melissa Rojas Yela

3.1.3 Resultados de desplazamientos con el programa.

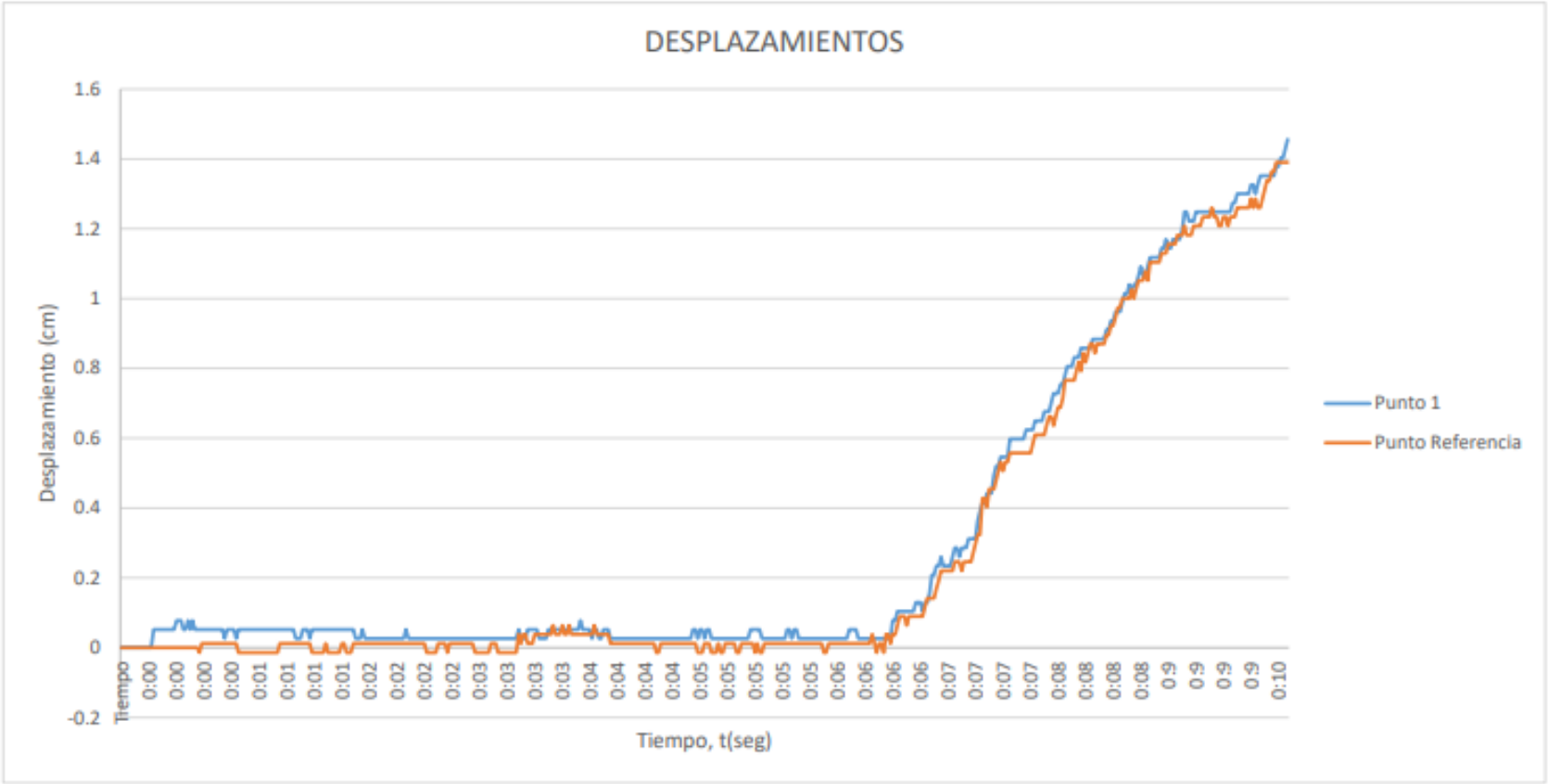


Gráfico 1 Respuesta de desplazamiento verticales del ensayo de flexión.

Elaborado por: Melissa Rojas Yela

3.1.4 Ensayo 2: Viga de madera balsa a corte.

Similar al ensayo de flexión, se usó soporte, con ambos extremos simplemente apoyados, pero con la única variación de que la carga puntual está a la distancia 2.5 del peralte de la sección ensayada con respecto a uno de los apoyos del elemento. Cabe destacar que se escogió esta distancia con el objetivo de ensayar el elemento a corte puro y determinando su desplazamiento.

Entonces con respecto a los ensayos a corte y que sea directo se ha realizado un ensayo de un elemento que fue colocada la fuerza a 2,5h de la viga.

A continuación, se muestra el ensayo a corte.

Ilustración 8 Elemento simplemente apoyado con una carga Puntual en uno de los extremos. Obtenido de: Melissa Rojas Yela

3.1.5 Propiedades del video.

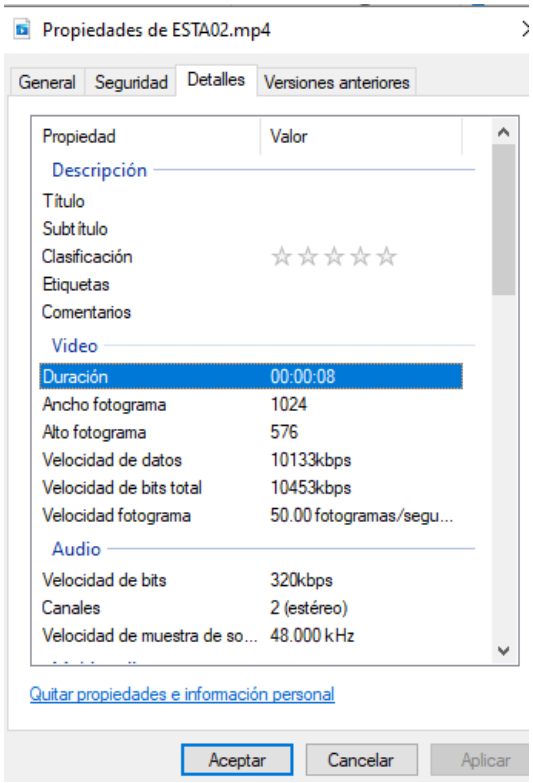


Ilustración 9 Propiedades del video de ensayo a corte.

Capturado por: Melissa Rojas Yela

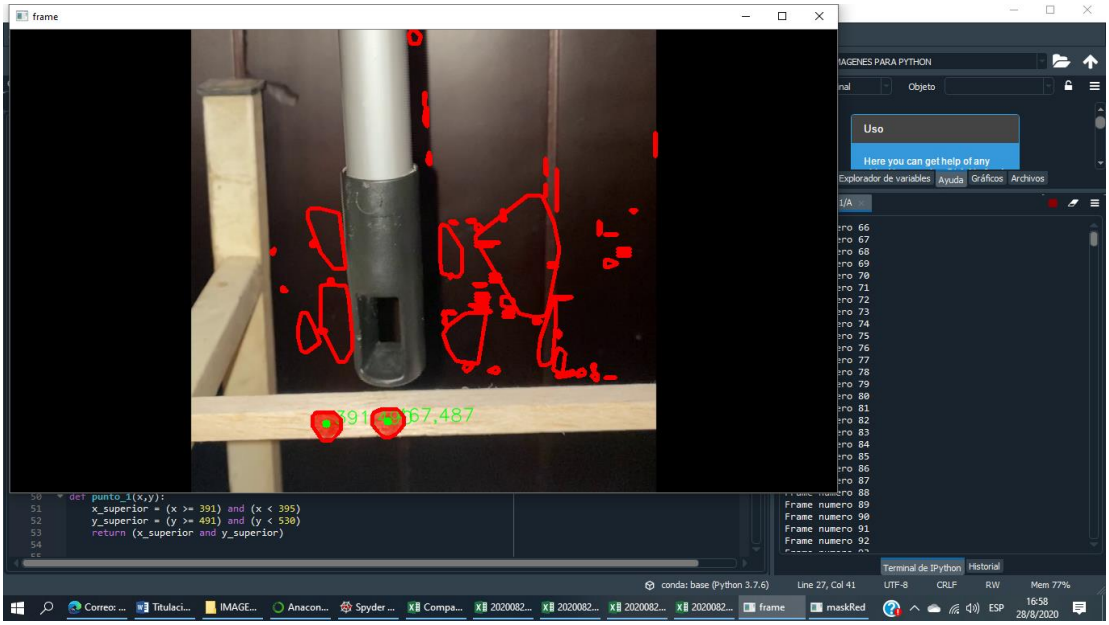


Ilustración 10 Lectura de coordenadas de pixeles en el programa.

Capturado por: Melissa Rojas Yela

3.1.6 Resultados de desplazamientos con el programa.

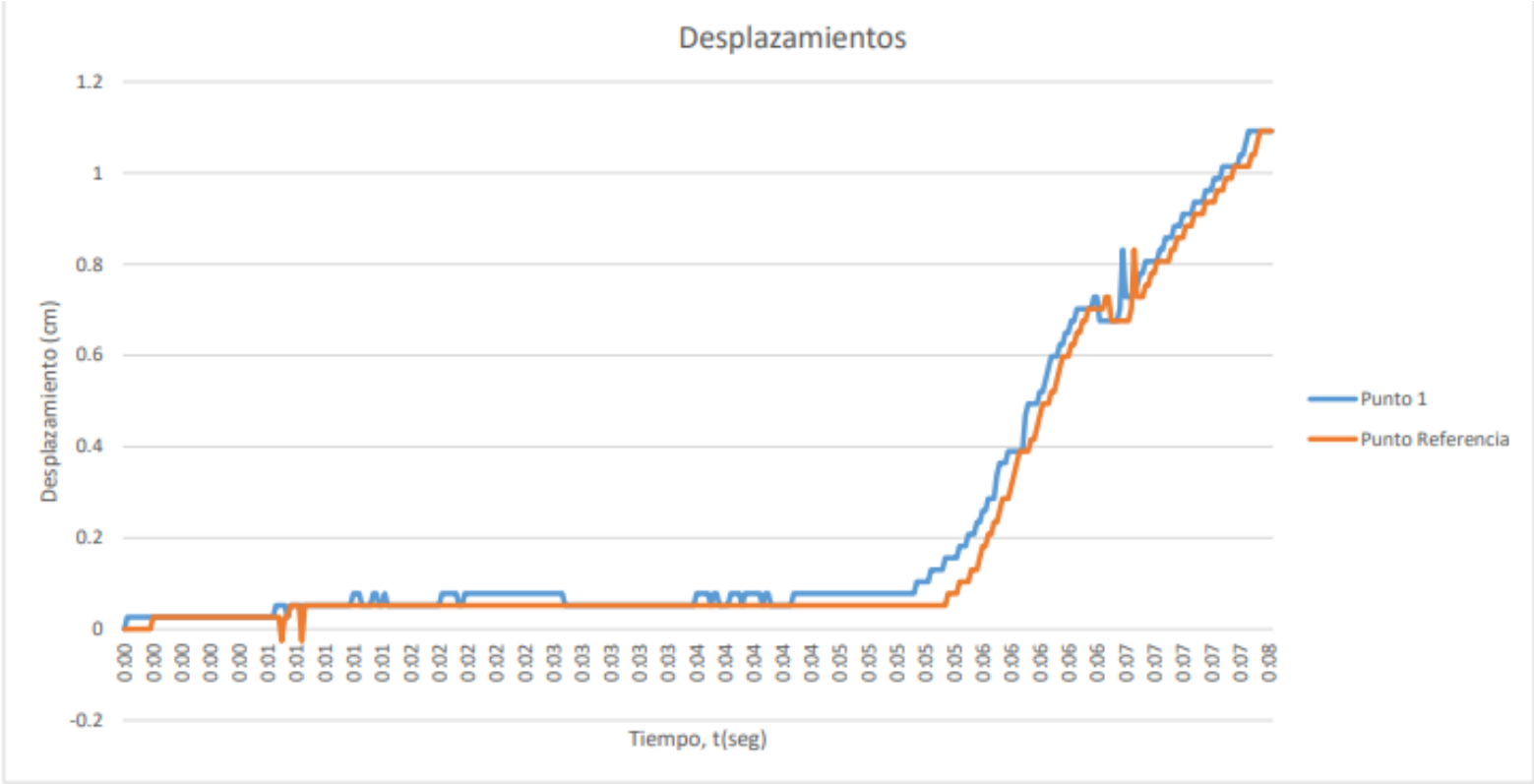


Gráfico 2 Respuesta de desplazamiento vertical en ensayo a corte.

Elaborado por: Melissa Rojas Yela

3.2 Ensayos dinámicos

3.2.1 Ensayo 1: A escala sobre mesa vibradora una columna de hormigón armado.

Selección del Modelo

Se selecciona este modelo para ser ensayado sobre mesa vibratoria que tiene las siguientes características:

Cuadro N° 4 Dimensiones y características del modelo

Propiedades	Modelo
Diámetro	35 cm
Altura	2. 40 m
Masa Superior	2m x 2m x 1.1m
Peso	11 Ton
Periodo	
Primer Modo	0.47 seg
Segundo Modo	0,7 seg

Elaborado por: Melissa Rojas Yela

Ilustración 11 Columna de Puente Escala del primer video

Capturado por: Melissa Rojas Yela

3.2.2 Propiedades del video.

Se analizó la lectura sacando dos partes del video para tener una mejor precisión de la movilización de los puntos.

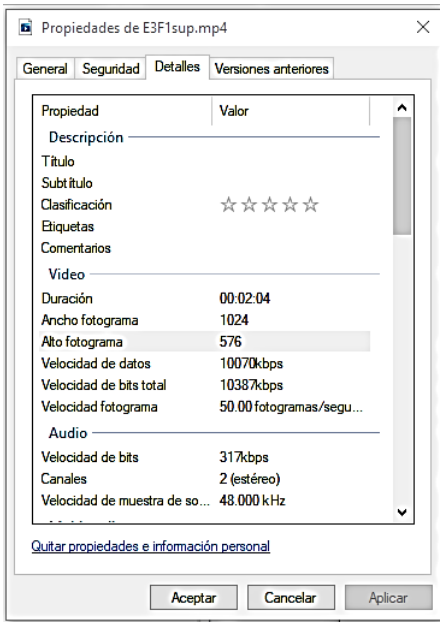


Ilustración 12 Propiedades del video de la parte superior del ensayo dinámico

Capturado por: Melissa Rojas Yela

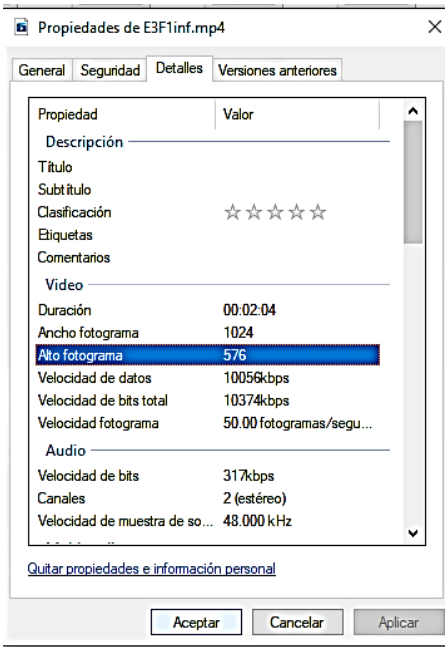


Ilustración 13 Propiedades del video de la parte inferior del ensayo dinámico.

Capturado por: Melissa Rojas Yela

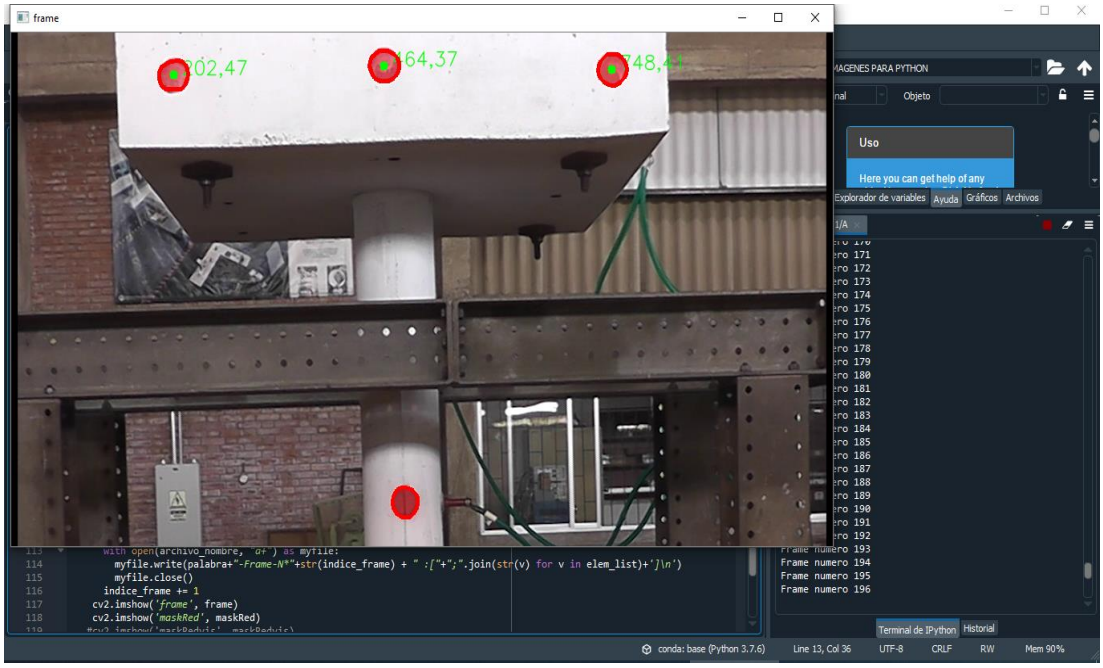


Ilustración 14 Parte superior del primer ensayo.

Capturado por: Melissa Rojas Yela

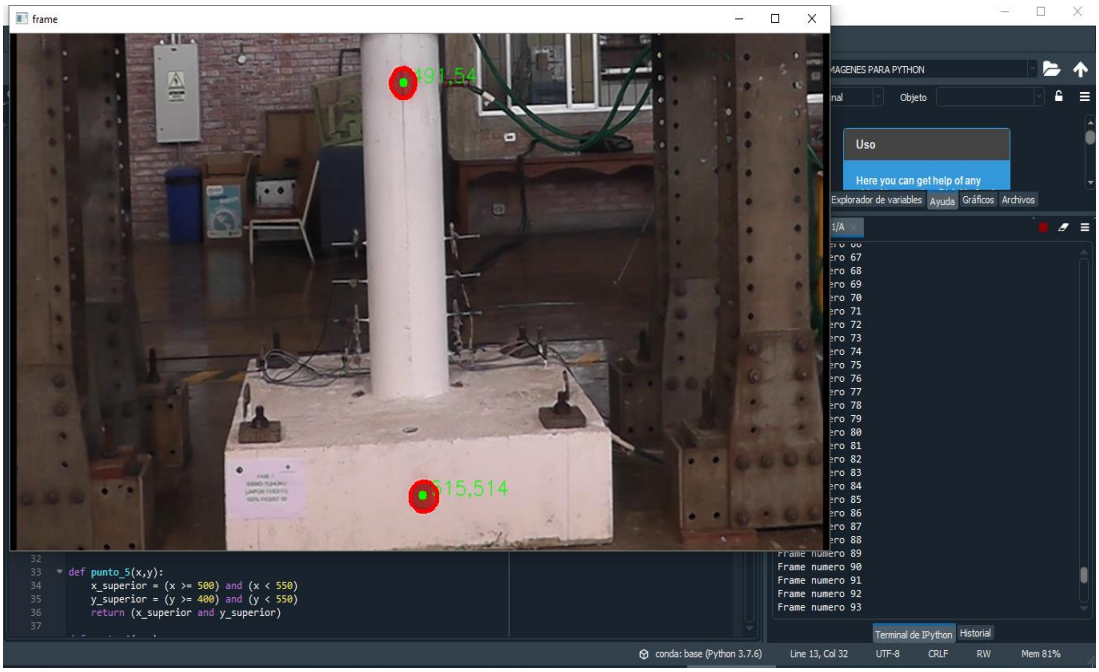


Ilustración 15 Parte inferior del primer ensayo

Capturado por: Melissa Rojas Yela

3.2.3 Resultado de desplazamientos con el programa.

Desplazamientos que fueron sacados con el programa del primer ensayo dinámico.

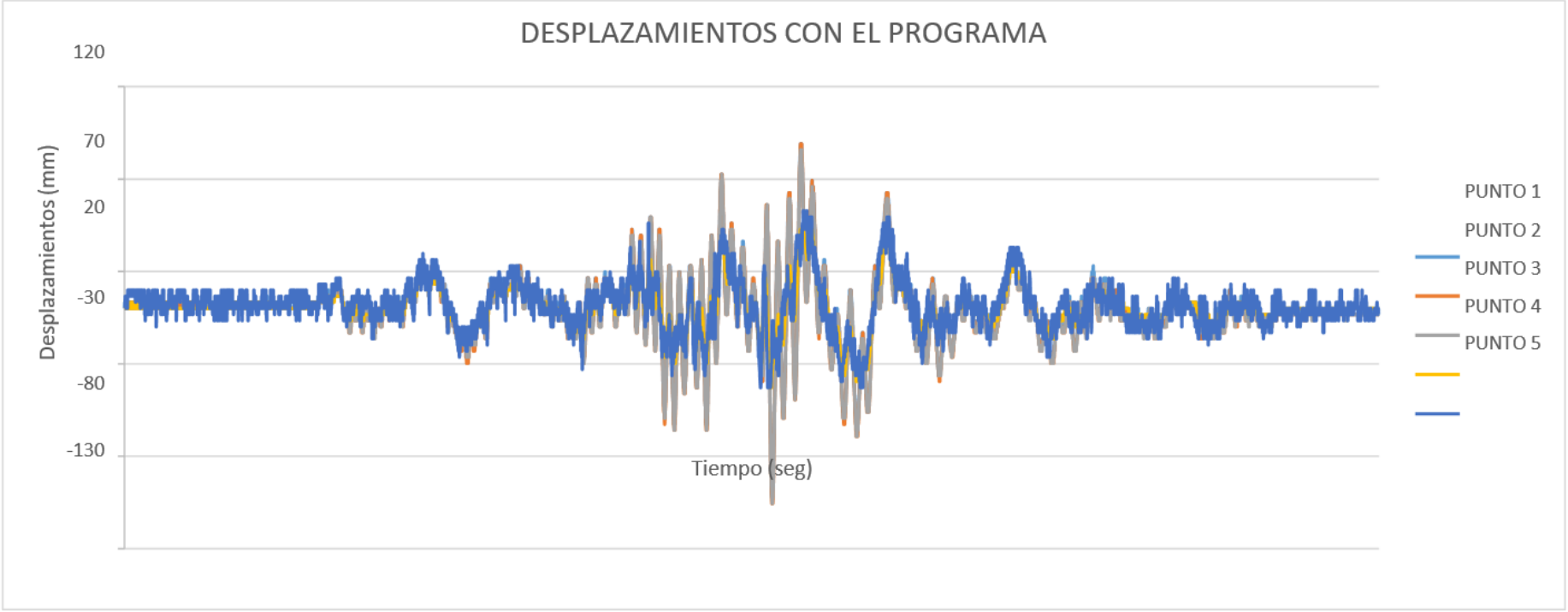


Gráfico 3 Resultados de desplazamientos del primer ensayo dinámico en base a la programación de Python.

Elaborado por: Melissa Rojas Yela

3.2.4 Resultado de desplazamientos con sensores.

1-2-3	4	5	correspondencia de puentos pintados en columna		
D2	D3	D7	correspondencia de sensores		

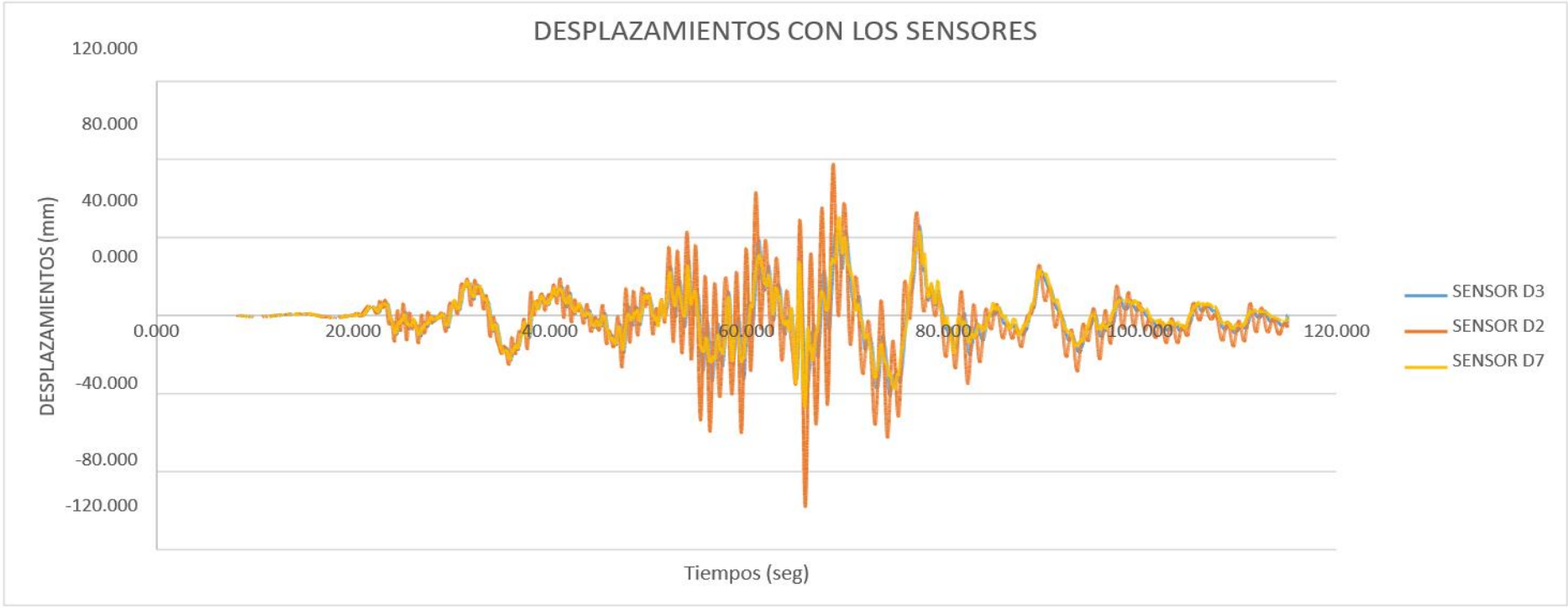


Gráfico 4 Resultados de desplazamientos del primer ensayo dinámico en base al seguimiento con sensores.

Obtenido por Tutor

3.2.5 Comparación entre el sensor D2 y del programa con el punto 2.

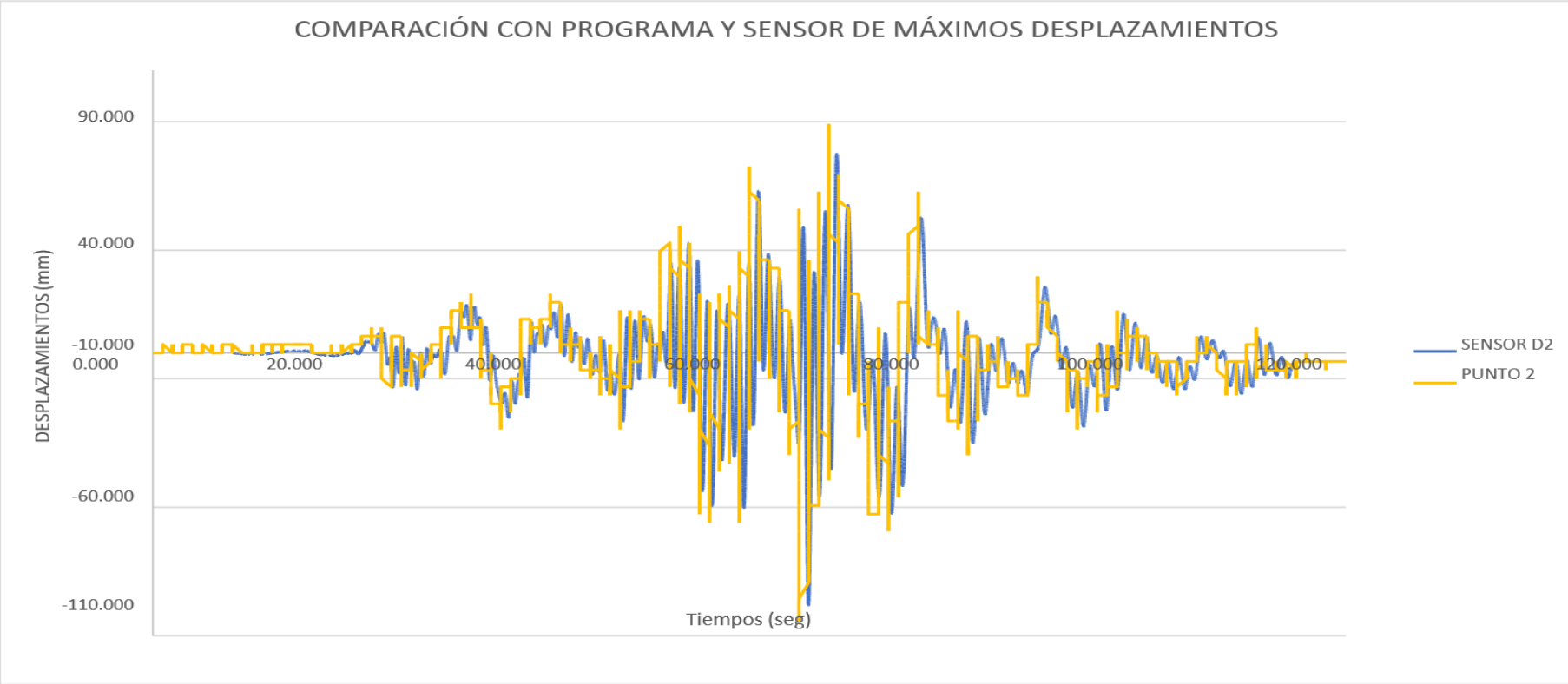


Gráfico 5 Comparación de desplazamientos del sensor D2 y punto 2.

Elaborado por: Melissa Rojas Yela

3.2.6 Ensayo 2: A escala sobre mesa vibradora una columna de hormigón armado.

Selección del modelo y prototipo

Se selecciona este modelo para ser ensayado sobre mesa vibratoria que tiene las siguientes características:

Cuadro N° 5 Dimensiones y características del modelo

Propiedades	Modelo
Diámetro	35 cm
Altura	2. 40 m
Masa Superior	2m x 2m x 1.1m
Peso	11 Ton
Periodo	
Primer Modo	0.8 seg
Segundo Modo	1,1 seg

Elaborado por: Melissa Rojas Yela

Ilustración 16 Columna de Puente escala del segundo video.

Capturado por: Melissa Rojas Yela

Se analizó la lectura sacando dos partes del video para tener una mejor precisión de la movilización de los puntos.

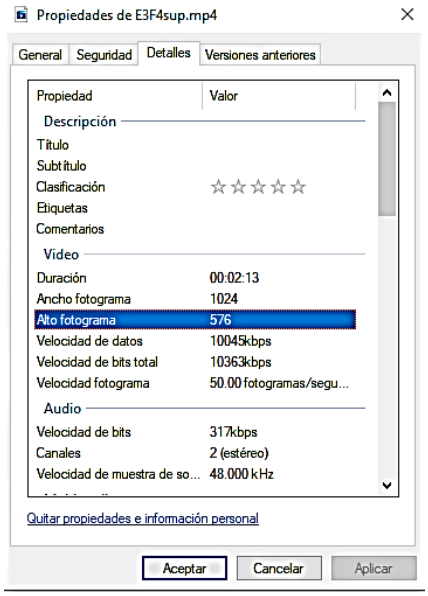


Ilustración 17 Propiedades del video de la parte superior del ensayo dinámico.

Capturado por: Melissa Rojas Yela

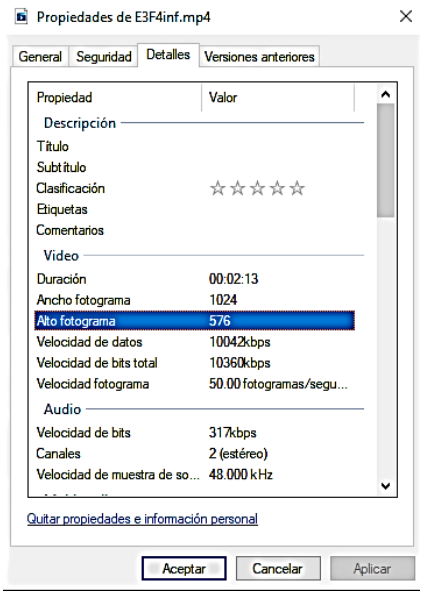


Ilustración 18 Propiedades del video de la parte inferior del ensayo dinámico.

Capturado por: Melissa Rojas Yela

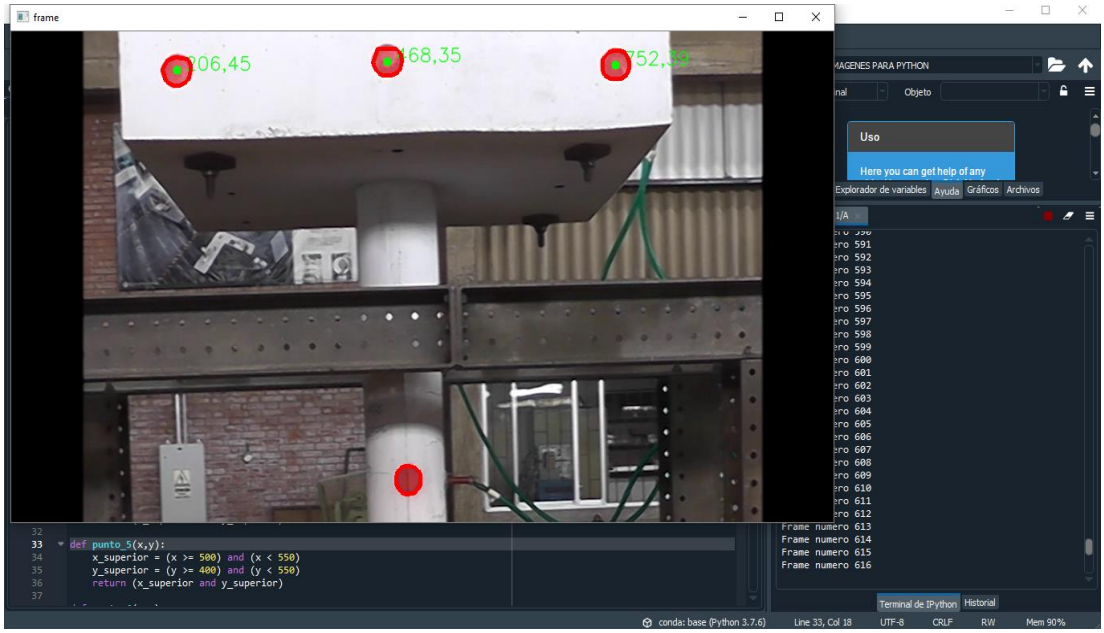


Ilustración 20 Parte superior del segundo ensayo.

Capturado por: Melissa Rojas Yela

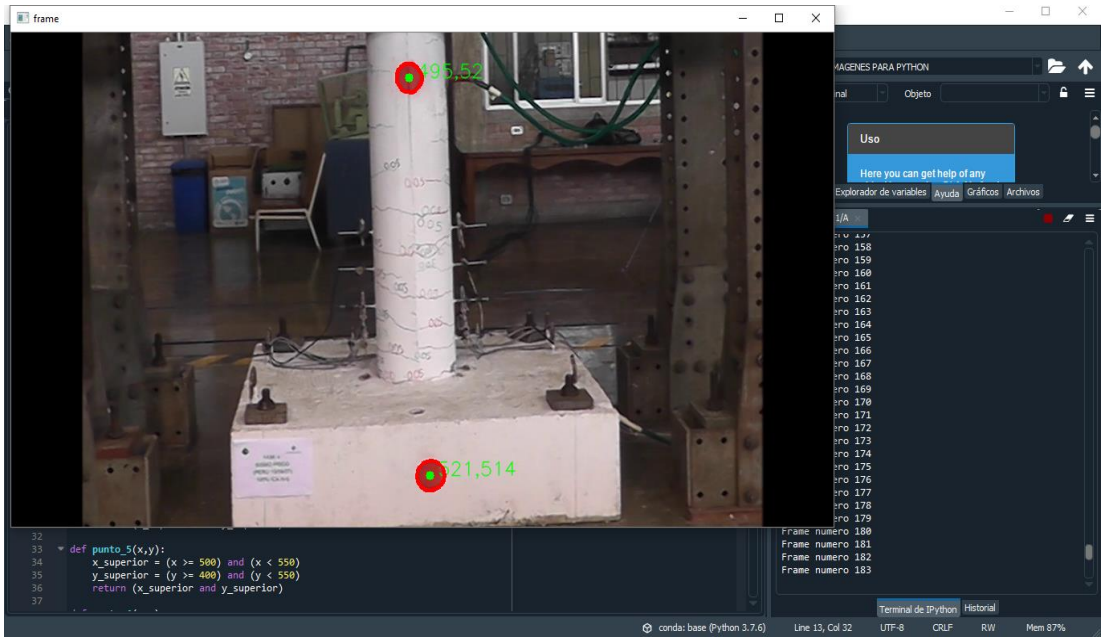


Ilustración 19 Parte inferior del segundo ensayo.

Capturado por: Melissa Rojas Yela

3.2.8 Resultado de desplazamientos con programa.

Desplazamientos que fueron sacados con el programa del segundo ensayo dinámico.

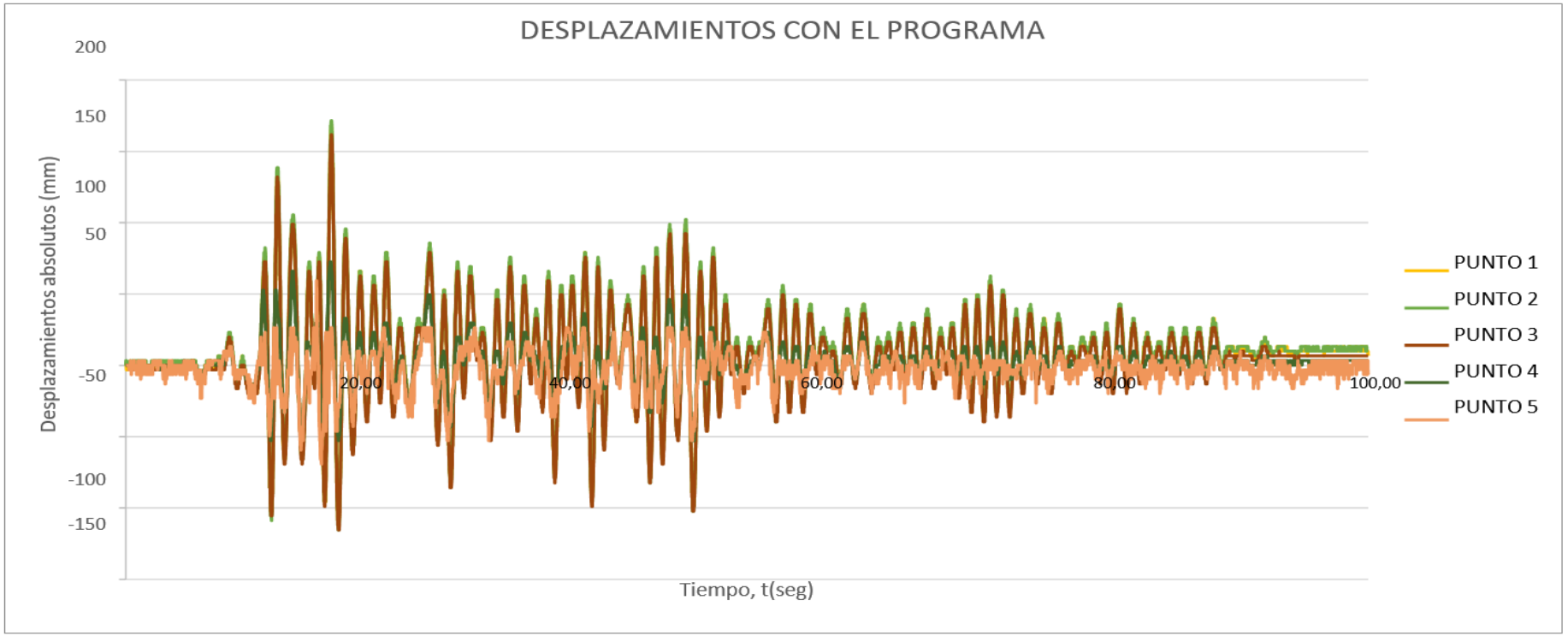


Gráfico 6 Resultados de desplazamientos del segundo ensayo dinámico en base a la programación de Python.

Elaborado por: Melissa Rojas Yela

3.2.7. Resultados de desplazamientos con sensores.

1-2-3	4	5	correspondencia de puentos pintados en columna		
D2	D3	D7	correspondencia de sensores		

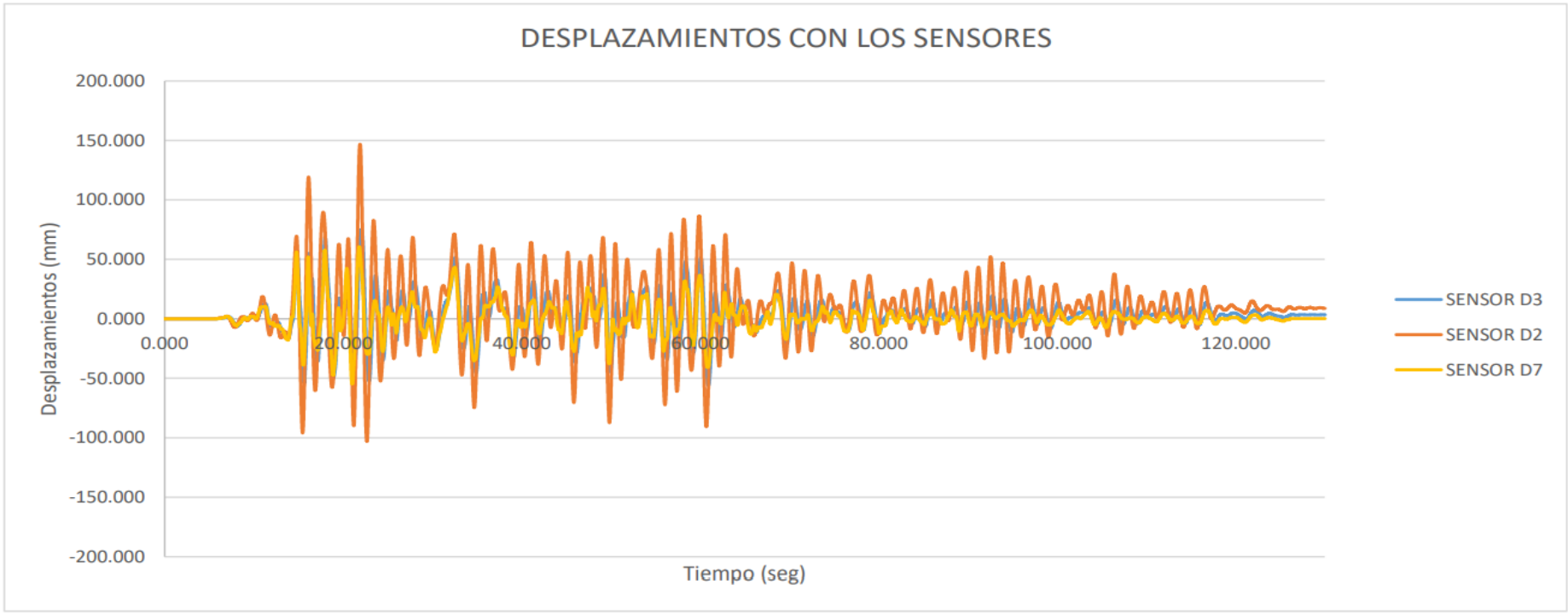


Gráfico 7 Resultados de desplazamientos del segundo ensayo dinámico en base al seguimiento con sensores.

Obtenido por Tutor

3.2.8. Comparación entre el sensor D2 y del programa con el punto 2

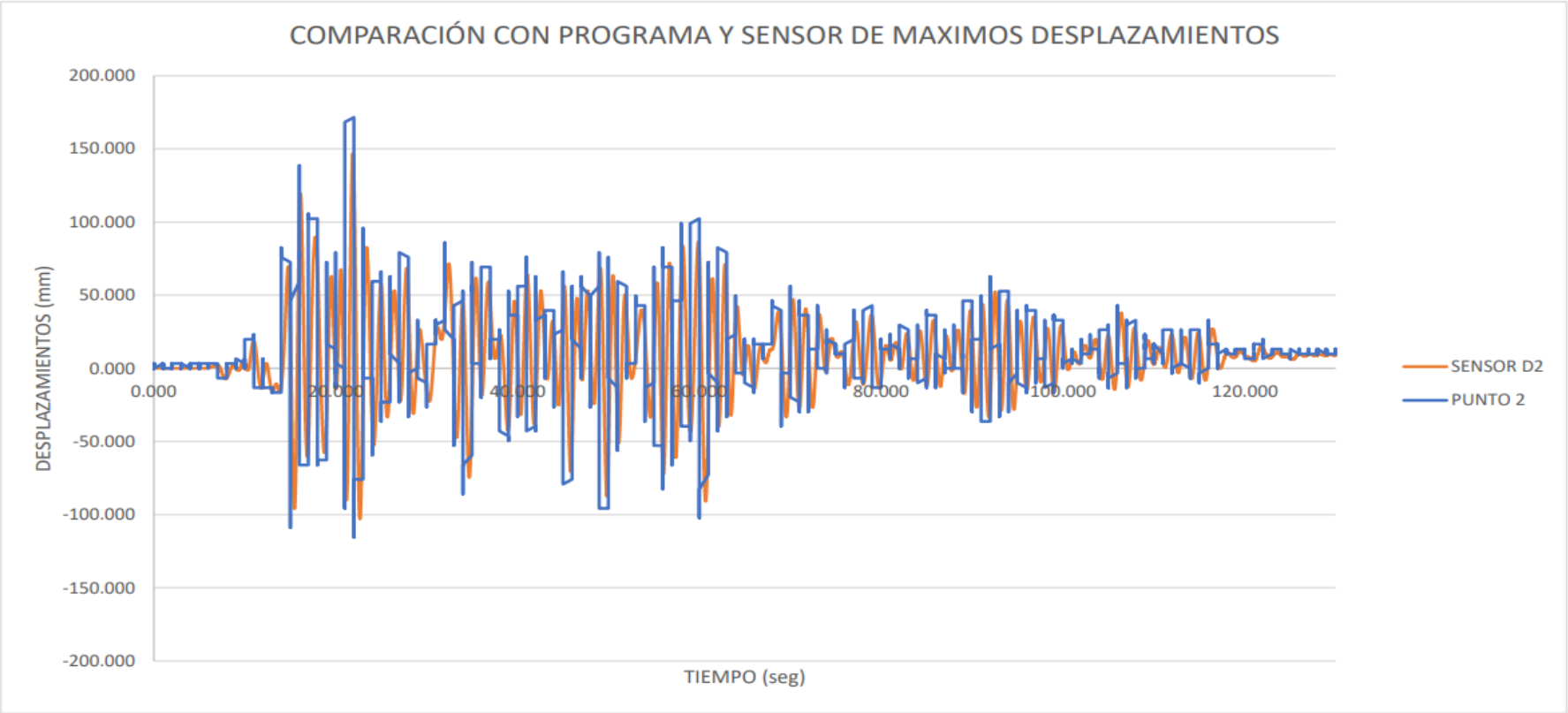


Gráfico 8 Comparación de desplazamientos del sensor D2 y punto 2

Elaborado por: Melissa Rojas Yela

3.2.9 Comparación de resultados.

Comparación de resultados de los máximos puntos de los 5 puntos.

Se va a comparar los resultados máximos de los ensayos dinámicos que se obtuvo usando los métodos de lectura con sensores y la programación con Python.

Cuadro N° 6 Comparación del primer ensayo dinámico de los valores máximos

Primer ensayo dinámico con programa			**Primer ensayo dinámico con sensor**		
Tiempo (seg)	64,000		**Tiempo (seg)**	65,965	Sensor
Desplazamiento (cm)	-10,23 -10,56 -10, 56	Puntos (1,2,3)	**Desplazamiento (cm)**	-9,79	D2 (Puntos 1,2,3)

Elaborado por: Melissa Rojas Yela

Cuadro N° 7 Comparación del segundo ensayo dinámico de los valores máximos

Segundo ensayo dinámico con programa			**Segundo ensayo dinámico con sensor**		
Tiempo (seg)	22,000		**Tiempo (seg)**	21,900	Sensor
Desplazamiento (cm)	16,83 17,16 16,17	Puntos (1,2,3)	**Desplazamiento (cm)**	14,687	D2 (Puntos 1,2,3)

Elaborado por: Melissa Rojas Yela

Interpretación del resultado

Se ha destacado los puntos máximos de desplazamientos en los ensayos dinámicos, notando que las mayores lecturas son en la parte superior del modelo contando con tres puntos, pese a esto el sensor D2 solo toma una lectura en general que incluye a los puntos 1,2,3. Mientras con el programa se obtiene las tres lecturas de los puntos 1, 2, 3.

Los puntos de control marcados en el ensayo dinámico con el color rojo tienen un diámetro de 10 cm.

Teniendo que, tanto en el primer ensayo dinámico como en el segundo, se ve la diferencia de resultados, se puede interpretar que la causa es la distancia de los puntos de lecturas entre ellos en el modelo y con respecto a la ubicación de la cámara.

En el modelo los puntos 1 y 3 tienen una distancia de 180 cm que fue tomada para realizar la equivalencia de cuanto tiene 1 pixel, mientras que en pixeles la distancia entre esos puntos es 546 pixeles dando que 1 pixel equivale a 0,329 cm. La distancia del espécimen a la cámara fue de 4 metros.

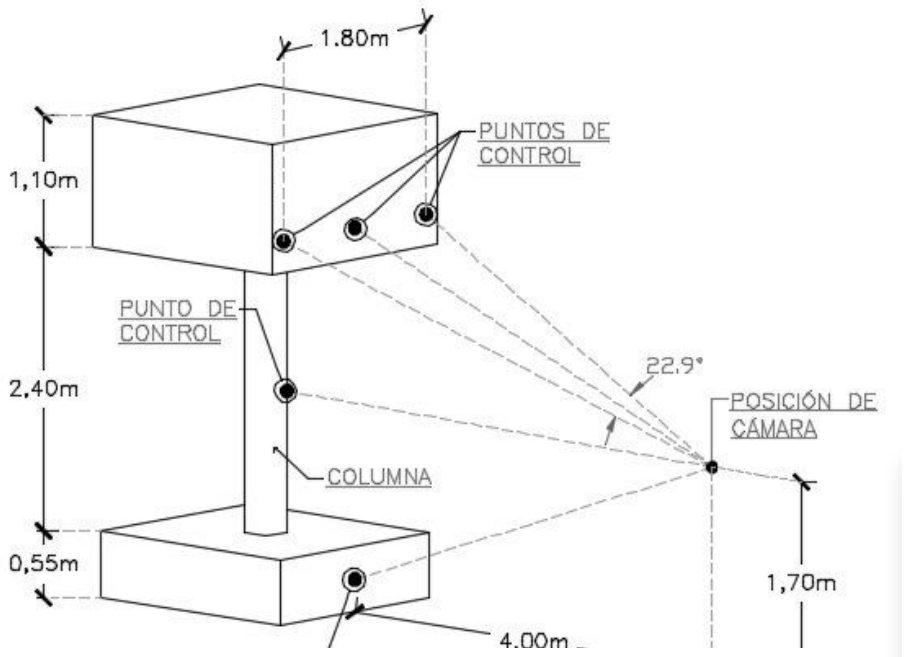


Ilustración 21 Demostración en perspectiva del ensayo dinámico

Elaborado por: Melissa Rojas Yela

Comparación de los máximos resultados en los puntos 4 y 5

Se compara de los puntos de control de la parte inferior del ensayo, se presenta valores máximos de desplazamientos de los puntos 4, 5 con el programa y los sensores D3, D7.

Cuadro N° 8 Comparación del primer ensayo dinámico de los puntos 4 y 5 con los valores máximos

<table>
<tr><th colspan="3">Primer ensayo dinámico con programa</th><th colspan="3">Primer ensayo dinámico con sensor</th></tr>
<tr><td>Tiempo (seg)</td><td>63,000</td><td rowspan="2">Punto 4</td><td>Tiempo (seg)</td><td>66,045</td><td rowspan="2">Sensor D3 (Punto 4)</td></tr>
<tr><td>Desplazamiento (cm)</td><td>-4,29</td><td>Desplazamiento (cm)</td><td>-4,6513</td></tr>
<tr><td>Tiempo (seg)</td><td>67,000</td><td rowspan="2">Punto 5</td><td>Tiempo (seg)</td><td>69,415</td><td rowspan="2">Sensor D7 (Punto 5)</td></tr>
<tr><td>Desplazamiento (cm)</td><td>5,28</td><td>Desplazamiento (cm)</td><td>4,9982</td></tr>
</table>

Elaborado por: Melissa Rojas Yela

Cuadro N° 9 Comparación del segundo ensayo dinámico de los puntos 4 y 5 con los valores máximos

Segundo ensayo dinámico con programa			Segundo ensayo dinámico con sensor		
Tiempo (seg)	21,000	Punto 4	**Tiempo (seg)**	22,050	Sensor D3 (Punto 4)
Desplazamiento (cm)	7,260		**Desplazamiento (cm)**	7,488	
Tiempo (seg)	21,000	Punto 5	**Tiempo (seg)**	21,800	Sensor D7 (Punto 5)
Desplazamiento (cm)	5,940		**Desplazamiento (cm)**	6,0727	

Elaborado por: Melissa Rojas Yela

Diferencia de resultados

Diferencia en los ensayos dinámicos con el programa y los sensores en los ensayos dinámicos.

Cuadro N° 10 Porcentajes de diferencia en el primer ensayo

Primer ensayo dinámico					
Programa		**Sensor**		**Diferencia**	**Porcentaje**
P1	-10,23	D2	-9,79	-0,44	4,49%
P2	-10,56		-9,79	-0,77	7,87%
P3	-10,56		-9,79	-0,77	7,87%
P4	-4,29	D3	-4,65	-0,36	8,39%
P5	5,28	D7	4,998	0,28	5,64%

Elaborado por: Melissa Rojas Yela

Cuadro N° 11 Porcentajes de diferencia en el segundo ensayo

Segundo ensayo dinámico					
Programa		**Sensor**		**Diferencia**	**Porcentaje**
P1	16,83	D2	14,69	2,14	14,57%
P2	17,16		14,69	2,47	16,81%
P3	16,17		14,69	1,48	10,07%
P4	7,26	D3	7,49	0,23	3,17%
P5	5,94	D7	6,07	0,13	2,19%

Elaborado por: Melissa Rojas Yela

En el segundo ensayo se ve la mayor diferencia entre los puntos llegando a un 16,81% y un 2,19% como mínimo de porcentajes de diferencias

Interpretación de resultados con los ensayos estáticos

Por otro lado, con los ensayos estáticos la distancia entre puntos es de 2cm y en pixeles es de 62pixeles dando la equivalencia que 1pixel es 0,032cm tomando en cuenta que al ser grabado el video para dichos ensayos la ubicación de la cámara fue aproximadamente de menos de 100 cm.

Los puntos de control que fueron realizados en el ensayo estático tienen un diámetro de 1cm de color rojo.

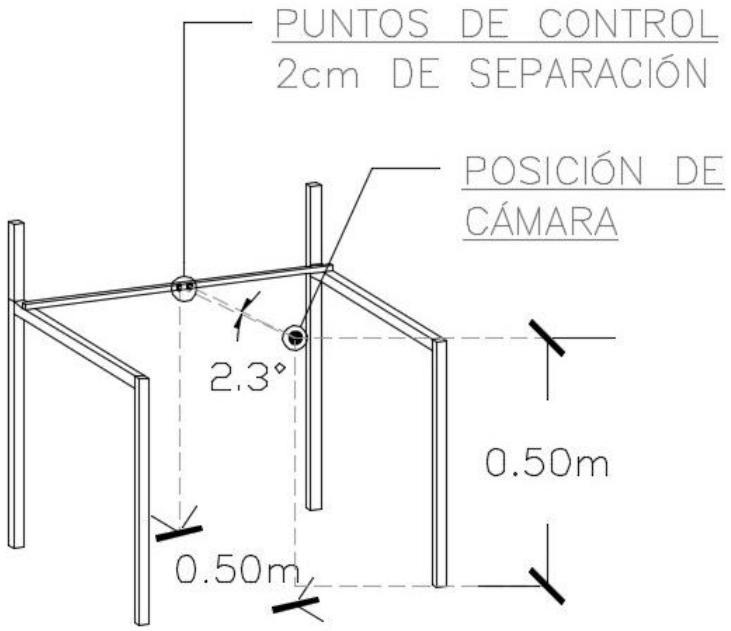


Ilustración 22 Demostración perspectiva del ensayo a flexión

Elaborado por: Melissa Rojas Yela

Ilustración 23 Demostración perspectiva del ensayo a corte

Elaborado por: Melissa Rojas Yela

CAPÍTULO IV

4 MANUAL DE USUARIO

4.1 Instalaciones

4.1.1 Python por medio de anaconda.

Revisar qué sistema operativo es el más adecuado, para entonces proceder a descargar el software de 64 o de 32 bits, en este caso se está usando el de 64 bits para la versión de Python 3.7.

- Ejecutar como administrador
- Realizar un acceso directo del programa
- Usamos Spyder

La dirección URL https://www.anaconda.com/distribution/ ayuda a descargar e instalar fácilmente y gratis el programa.

4.1.2 OpenCV en Python

OpenCv en su visión por computadora se ha vuelto muy popular para análisis y tratamiento de imágenes con uso de algoritmos de inteligencia artificial, Uno de los usos más importantes es en fotografía y en marketing, para la detección de rostros y objetos, también se usa en seguridad. OpenCV hoy es la biblioteca de visión por computadora más grande, sobre todo por las diversas funciones de aplicación, es gratuita para fines comerciales y académicos, cuenta con más de 2500 algoritmos, ya está dicho que es una gran biblioteca con interfaces para diversos lenguajes como Python, Java y C++. es compatible con los sistemas, Windows, Mac OS y Linux, para aprovechar mejor esta gran librería, hay que tener conocimientos de: Librería Numpy y Librería Matplotlib. OpenCV se puede instalar en Python desde archivos binarios y archivos fuente previamente compilados, según la plataforma que ha de usarse, Windows y Mac OS; o mediante los paquetes estándar de Windows, MacOS y casi cualquier distribución GNU/Linux: si solo se necesita

módulos principales se debe ejecutar *pip install opencv-python, en cambio,* si se necesitan los módulos principales y los adicionales (contrib) se debe ejecutar *pip install opencv-contrib-python,* estos comandos pueden escribirse en Jupyter o cualquier otro IDE. (Marín, 2020).

4.2 Manual, obtención de desplazamientos en Python

Elaborado por Rojas Yela, Melissa Michelle

Presentación

Al mostrar este trabajo se requiere que sea una de las mayores herramientas para los docentes y estudiantes consiguiendo mejorar la parte experimental de las diferentes materias que ofrece la Facultad de Ingeniería.

Se elaborará con el fin que sea sencillo para el uso, con diferentes tipos de ensayos y sugerencia que se pueda dar a través del manual.

De igual manera puede ser útil para aclarar algunos conceptos sobre desarrollo de alternativa de medición de desplazamientos en ensayos de laboratorio empleando técnicas de procesamiento de imágenes.

Es importante indicar que solo comprende técnicas de procesamiento de imágenes empleando el programa Python.

Detalle

Para el procesamiento de imágenes se debe tener en cuenta que se puede procesar de dos formas, sea por lectura de imagen o de un video.

En este trabajo se basará en el procesamiento de video por medio de Python usando Spyder que es un entorno de códigos abierto que se maneja con lenguaje Python, para la visualización de imagen real y que se dé como final respuesta de información numérica se va a importar Opencv, para elaboración de matrices, dominio de álgebra lineal y cálculos matemáticos números se va a importar NumPy.

Procesamiento de imagen

Capturar un video archivado

Con este paso se va a llevar la visualización del video original con el que se va a seguir el proceso del manual, para más enfoques que se requiera para llegar a una respuesta.

```
1   import cv2
2
3   cap = cv2.VideoCapture("ARMONICOP2.mp4")
4   while True:
5     ret, frame = cap.read()
6     if ret == True:
7       cv2.imshow('frame', frame)
8       if cv2.waitKey(30)& 0xFF== ord("s"):
9         break
10
11  cap.release()
12  cv2.destroyAllWindows()
13
```

Ilustración 24 Programación de captura de imagen

Elaborado por: Melissa Rojas Yela

Se empieza la explicación de cada una de las filas con los códigos a utilizar para un mejor aprendizaje y desempeño que se le pueda dar.

- Fila 1 Se importa OpenCv
- Fila 3 Para no tener que escribir en cada momento la función **"cv2.VideoCapture()"** se le ha asignado una variable sencilla, dicha función ayudará a procesar captura del video.

El nombre dentro del paréntesis es el archivo que se procesará, bien se puede poner el nombre como tal del documento seguido del tipo de archivo o escribir la ruta de donde se encuentre grabado, si no es un archivo guardado se puede grabar con la misma cámara del ordenador colocando 0 o -1.

- Fila 4 que lo que se escriba en las siguientes filas se procesará solo si es verdadero.
- Fila 5 Aquí se ven dos variables por motivo de la función **".read()",** uno es una variable booleana, mientras la otra es la imagen en sí. Siendo:

ret: si se ha capturado una imagen es True, mientras es False cuando no se ha capturado una imagen.

Frame: es dicha imagen que se va a trabajar.

- Fila 6 Va a permitir seguir si se ha capturado una imagen con la condición **“if ret == True”**.
- Fila 7 Indica la visualización de la imagen donde se va a dar un nombre de la ventana que se va abrir, siguiendo la variable que ya fue realizada que es la imagen que fue capturada para ser procesada.
- Fila 8 Va ayudar a visualizar esta condición hasta permitir salir del programa o terminar la visualización, teniendo que presionar la tecla “s”. con & 0xFF, cuando el ordenador es de 64bits que es el programa en el que se está trabajando también. Viendo que **“cv2.waitkey”** indica un número entre paréntesis siendo el 30 que va a demostrar el video con la velocidad del mismo, si se coloca 1 esto el programa considerará ir más rápido sin importar en fps original del video.
- Fila 9 Se va activar si se realiza la condición de presionar la s para cerrar la ventana saliéndose del While.
- Fila 11 Captura del video es liberada.
- Fila 12 Cierra la visualización.

Detección de colores con HSV en OpenCV del video archivado

Se lo realiza para una mayor facilidad ya que OpenCV lee en BGR las imágenes, para eso se usará función que permita la transformación y se pueda procesar lo que se quiere.

```
1   import cv2
2
3   cap = cv2.VideoCapture("ARMONICOP2.mp4")
4   while True:
5     ret, frame = cap.read()
6     if ret == True:
7       frameHSV = cv2.cvtcolor(frame,cv2.COLOR_BGR2HSV)
8       cv2.imshow('frame', frame)
9       if cv2.waitKey(30)& 0xFF== ord("s"):
10        break
11
12  cap.release()
13  cv2.destroyAllWindows()
```

Ilustración 25 Programación de transformación de BGR a HSV

Elaborado por: Melissa Rojas Yela

La nueva fila ayudará a la transformación que se quiere, se colocará el código con la función dentro del bucle que siendo así se realice solo si se captura la imagen.

- Fila 7 La variable que se ha creado es por la función **"cv2.cvtcolor()"** colocando la imagen a transformar seguido de "cv2.COLOR_BGR2HSV" indicando la transformación**.**

Como siguiente se debe introducir el rango del color que se quiera detectar, basándose en el color rojo de igual manera se dejará los rangos de los colores amarillo y azul por si un ensayo es detectado dos colores se los pueda procesar sin tener problema alguno, con la ilustración 1 se puede seguir sacando rango de los otros colores siendo el que necesites por motivo de algún video guardado que obtenga otro color, o se esté realizando un video con diferente color siendo así, la ilustración 1 ayudará a seguir.

Rangos

Para realizar la detección con el color rojo en el programa se maneja con HSV en el color existen dos rangos, lo cual se lo puede evidenciar en la imagen 1.

rojoBajo1 = np.array ([0, 100, 20], np.uint8)
rojoAlto1 = np.array ([8, 255, 255], np.uint8)
rojoBajo2 = np.array ([175, 100, 20], np.uint8)
rojoAlto2 = np. array ([179, 255, 255], np.uint8)

Se puede usar los dos rangos o solo el rango que vaya a favorecer en el procesamiento del color para una mejor precisión de detección del color

azulBajo = np.array([100,100,20],np.uint8)
azulAlto = np.array([125,255,255],np.uint8)

amarilloBajo = np.array([15,100,20],np.uint8)
amarilloAlto = np.array([45,255,255],np.uint8)(Solano, 2019b)

```
1   import cv2
2   import numpy as np
3
4   cap = cv2.VideoCapture("ARMONICOP2.mp4")
5   rojoBajo1 = np.array([0, 100, 20], np.uint8)
6   rojoAlto1 = np.array([5, 255, 255], np.uint8)
7   rojoBajo2=np.array([175, 100, 20], np.uint8)
8   rojoAlto2=np.array([180, 255, 255], np.uint8)
9   while True:
10    ret, frame = cap.read()
11    if ret == True:
12      frameHSV = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
13      maskRojo1 = cv2.inRange(frameHSV, rojoBajo1, rojoAlto1)
14      maskRojo2 = cv2.inRange(frameHSV, rojoBajo2, rojoAlto2)
15      maskRojo = cv2.add(maskRojo1, maskRojo2)
16      cv2.imshow('frame', frame)
17      cv2.imshow('maskRojo', maskRojo)
18      if cv2.waitKey(30)& 0xFF== ord("s"):
19        break
20
21  cap.release()
22  cv2.destroyAllWindows()
23
```

Ilustración 26 Programación de rangos de colores

Elaborado por: Melissa Rojas Yela

Para este proceso del rango del color se debe importar NumPy que ayudará a colocar la matriz de los rangos de los colores siendo así, se procesará.

- Fila 5 y 6 se está dando los primeros rangos del color rojo, introduciendo un array con la librería NumPy, ya sabiendo que el rango está siendo leído como H, S y V seguido de “np.uint8”. Siendo lo mismo para el rango 2, ubicando antes del bucle debido a que solo será emitidas una vez.

- Fila 13, 14 y 15 Será para poder procesar los rangos, usando la función “cv2.inRange” colocando la imagen que va a ser procesada, ya si los límites de cada uno de los rangos.

Las variables maskRojo1 y maskRojo2 es imágenes binarias que mostrará donde se encuentra el color detectando y tener presente para realizar si se requiere cuadriculas imaginarias para que dicho color solo sea encontrado en el sitio que se quiere.

Si bien se tiene dos rangos los mismos que se deben unir, si solo se usa un solo rango no se uniría en el seguimiento de generar los códigos, para aquello se realizó la Fila 15 para que ambas imágenes sean mostradas una sola.

- Fila 16 Visualización del video de cómo se vería dándole nombre “frame”

Ilustración 27 Captura de video en transmisión
Elaborado por: Melissa Rojas Yela

- Fila 17 Visualizacion del maskRojo

Ilustración 28 Captura de la imagen binaria luego de ver detección del color rojo

Elaborado por: Melissa Rojas Yela

Se muestra la deteccion del color rojo donde el programa a visualizado el color por el rango que se ha colocado, se nota el color blanco de donde ha sido la deteccion y es donde se va a fijar más adelante en los puntos que se quiera para después tener seguimiento de coordenadas.

Una vez que se ha podido detectar el color damos seguimiento para obtener coordenadas de dicho color.

```python
import cv2
import numpy as np

cap = cv2.VideoCapture("ARMONICOP2.mp4")
rojoBajo1 = np.array([0, 100, 20], np.uint8)
rojoAlto1 = np.array([5, 255, 255], np.uint8)
rojoBajo2=np.array([175, 100, 20], np.uint8)
rojoAlto2=np.array([180, 255, 255], np.uint8)
while True:
  ret, frame = cap.read()
  if ret == True:
    frameHSV = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    maskRojo1 = cv2.inRange(frameHSV, rojoBajo1, rojoAlto1)
    maskRojo2 = cv2.inRange(frameHSV, rojoBajo2, rojoAlto2)
    maskRojo = cv2.add(maskRojo1, maskRojo2)
    contornos,_ = cv2.findContours(maskRojo, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    cv2.drawContours(frame, contornos, -1, (0,0,255), 3)
    cv2.imshow('frame', frame)
    cv2.imshow('maskRojo', maskRojo)
    if cv2.waitKey(30)& 0xFF== ord("s"):
      break

cap.release()
cv2.destroyAllWindows()

```

Ilustración 29 Programación de contornos

Elaborado por: Melissa Rojas Yela

Se encuentra las funciones para los contornos que por el momento señalará los que se detecte.

- Fila 16 La variable contornos,_ se manejará una función que una vez que se tenga la imagen binarizada se pueda dar la función “cv2.findContours” dando que coja la imagen que se realizón binaria, seguido de cv2.RETR_EXTERNAL para que tome en cuenta los contornos externos y como final cv2.CHAIN_APPROX_SIMPLE que comprime segmentos horizontales, verticales y diagonales dejando los puntos que se quiere.
- Fila 17 La función “cv2.drawContours” dibujará los contornos en frame, especifica los contornos, con -1 ya serian todos los contornos que se van encontrando, se debe especificar que color encerrará el objeto encontrado y una vez con detallar el color que va a ser el contorno se da el grosor.

Dando como resultado la ilustración siguiente

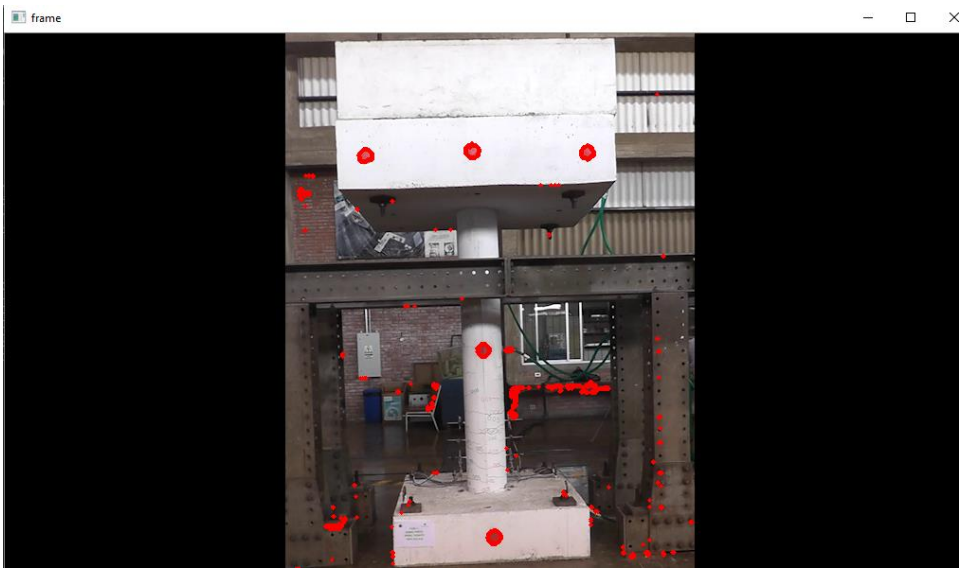


Ilustración 30 Captura de contornos dibujados en frame
Elaborado por: Melissa Rojas Yela

Como se puede ver en la ilustración 30 se detecta todos los puntos que estan dentro del rango que ha creado para la detección del color y realice los contornos, por el motivo que no se quiere todos los puntos solo las

coordenadas de 5 puntos, el dibujar contorno de acuerdo a su área se realizará.

```
1   import cv2
2   import numpy as np
3
4   cap = cv2.VideoCapture("ARMONICOP2.mp4")
5   rojoBajo1 = np.array([0, 100, 20], np.uint8)
6   rojoAlto1 = np.array([5, 255, 255], np.uint8)
7   rojoBajo2=np.array([175, 100, 20], np.uint8)
8   rojoAlto2=np.array([180, 255, 255], np.uint8)
9   while True:
10    ret, frame = cap.read()
11    if ret == True:
12      frameHSV = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
13      maskRojo1 = cv2.inRange(frameHSV, rojoBajo1, rojoAlto1)
14      maskRojo2 = cv2.inRange(frameHSV, rojoBajo2, rojoAlto2)
15      maskRojo = cv2.add(maskRojo1, maskRojo2)
16      contornos,_ = cv2.findContours(maskRojo, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
17      cv2.drawContours(frame, contornos, -1, (0,0,255), 3)
18      for c in contornos:
19        area = cv2.contourArea(c)
20        if area > 80:
21          M = cv2.moments(c)
22          if (M["m00"]==0): M["m00"]=1
23          x = int(M["m10"]/M["m00"])
24          y = int(M['m01']/M['m00'])
25          cv2.circle(frame, (x,y), 7, (0,255,0), -1)
26      cv2.imshow('frame', frame)
27      cv2.imshow('maskRojo', maskRojo)
28      if cv2.waitKey(30)& 0xFF== ord("s"):
29        break
30
31  cap.release()
32  cv2.destroyAllWindows()
33
```

Ilustración 31 Programación de coordenadas de pixeles

Elaborado por: Melissa Rojas Yela

- Fila 18 Se analiza cada contorno “c” , con un “for” se va recorriendo cada uno de los contornos encontrados.
- Fila 19 Se crea una variable para emplear la función “cv2.contourArea”, para así tener el área en pixeles del contorno.
- Fila 20 Se compara para este caso con 80 pixeles, para que solo pase los contornos que sean mayor a ese valor, siendo descartados los más pequeños, puede ser variable ese valor dependiendo del ditanciamiento de la toma del video.
- Fila 21 Variable en la que la función “cv2.moments” encuentra los momentos de los contornos, siendo así que ayudará a encontrar los puntos centrales tanto en X e Y.
- Fila 22 Se realiza una condición debido a la división que se basa y para que no exista división para cero, se lo iguala a 1
- Fila 23 y 24 Se obtiene los centroides de X e Y.

- Fila 25 En frame con “cv2.circle” se dibujará un circulo, en los puntos que van siendo encontrados con coordenadas X e Y con radio 7 pixeles, de color verde, colocando -1 por ser un circulo y no dibuje circunferencia.

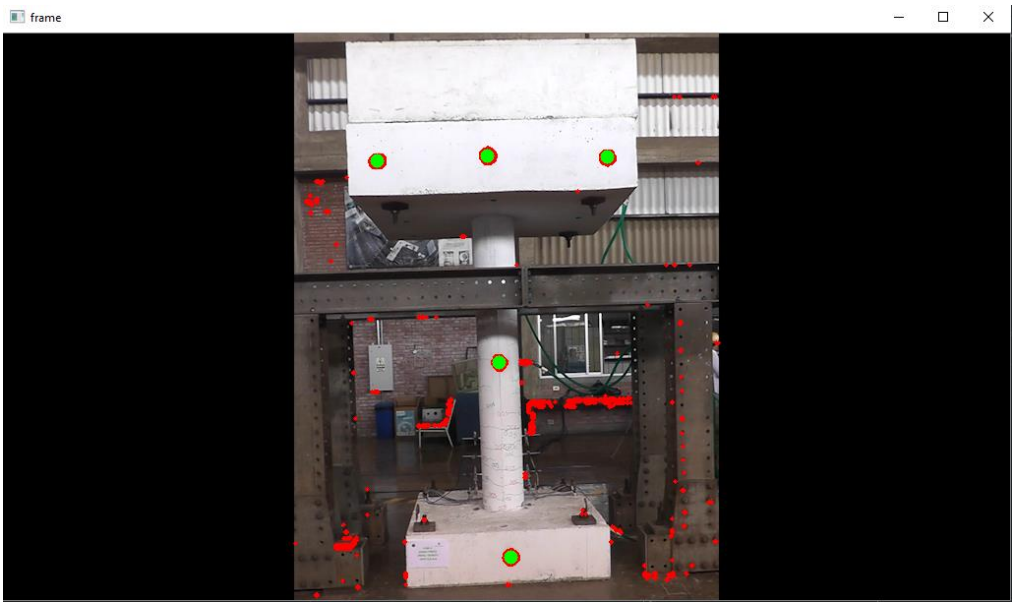


Ilustración 32 Captura de los círculos verdes para de detección de coordenadas
Elaborado por: Melissa Rojas Yela

Por motivos de contornos que se observan extras se realizará nuevo contorno para ser más precisos con los puntos que se requieren, y poder sobresalir los puntos necesarios.

```python
import cv2
import numpy as np

cap = cv2.VideoCapture("ARMONICOP2.mp4")
rojoBajo1 = np.array([0, 100, 20], np.uint8)
rojoAlto1 = np.array([5, 255, 255], np.uint8)
rojoBajo2=np.array([175, 100, 20], np.uint8)
rojoAlto2=np.array([180, 255, 255], np.uint8)
while True:
  ret, frame = cap.read()
  if ret == True:
    frameHSV = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    maskRojo1 = cv2.inRange(frameHSV, rojoBajo1, rojoAlto1)
    maskRojo2 = cv2.inRange(frameHSV, rojoBajo2, rojoAlto2)
    maskRojo = cv2.add(maskRojo1, maskRojo2)
    contornos,_ = cv2.findContours(maskRojo, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    for c in contornos:
      area = cv2.contourArea(c)
      if area > 80:
        M = cv2.moments(c)
        if (M["m00"]==0): M["m00"]=1
        x = int(M["m10"]/M["m00"])
        y = int(M['m01']/M['m00'])
        cv2.circle(frame, (x,y), 7, (0,255,0), -1)
        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(frame, '{},{}'.format(x,y),(x+10,y),font, 0.75,(0,255,0),1,cv2.LINE_AA)
        nuevoContorno = cv2.convexHull(c)
        cv2.drawContours(frame, [nuevoContorno], 0, (255,0,0), 3)
    cv2.imshow('frame', frame)
    cv2.imshow('maskRojo', maskRojo)
    if cv2.waitKey(30)& 0xFF== ord("s"):
      break

cap.release()
cv2.destroyAllWindows()
```

Ilustración 33 Programación de visualización de texto

Elaborado por: Melissa Rojas Yela

- Fila 26 Para visualizar el texto, se basará con la función “cv2.FONT_HERSHEY_SIMPLEX” para así usar la función “cv2.putext”
- Fila 27 Se realiza para poder ubicar el texto en la imagen con la que se está trabajando, la ubicación de las coordenadas y la fuente que de la variable Font, el tamaño, el color en BGR, con un grosor de línea que en este caso se usa 3.
- Fila 28 Aquí la función corrige si existiera un defecto de convexidad.
- Fila 29 No se va a visualizar ya las partículas que salían anteriormente que no se necesitaba.

Ilustración 34 Captura de los puntos que se requieren con las coordenadas de pixeles. Elaborado por: Melissa Rojas Yela

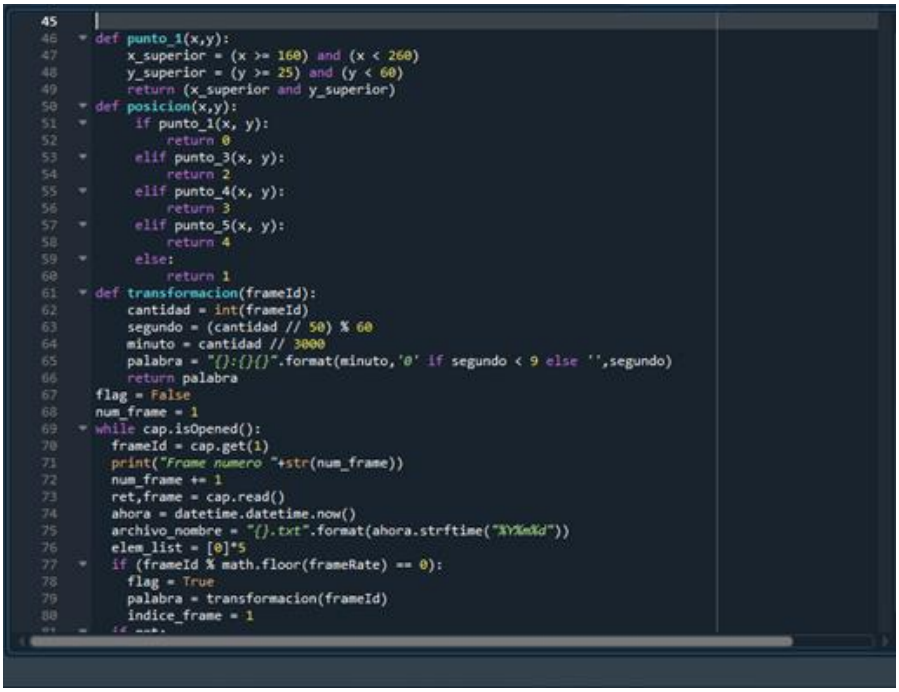
Se realiza uso de librerías nativas de Python que son datetime el cual cuenta con módulos para manejo del tiempo y math para hacer uso de funciones matemáticas.

```
9   import cv2
10  import numpy as np
11  import datetime
12  import math
13  cap = cv2.VideoCapture("E3F1sup.mp4")
14  frameRate = cap.get(5)
15  redBajo2=np.array([175, 100, 20], np.uint8)
16  redAlto2=np.array([180, 255, 255], np.uint8)
17  ahora = datetime.datetime.now()
18  archivo_nombre = "{}.txt".format(ahora.strftime("%Y%m%d"))
19  open(archivo_nombre,"w+").close()
20
21  def cuadricula_superior(x,y):
22      x_superior = (x >= 160) and (x < 810)
23      y_superior = (y >= 25) and (y < 60)
24      return (x_superior and y_superior)
25
26  def cuadricula_inferior(x,y):
27      x_superior = (x >= 500) and (x < 550)
28      y_superior = (y >= 400) and (y < 550)
29      return (x_superior and y_superior)
30
31  def punto_5(x,y):
32      x_superior = (x >= 500) and (x < 550)
33      y_superior = (y >= 400) and (y < 550)
34      return (x_superior and y_superior)
35
36  def punto_4(x,y):
37      x_superior = (x >= 440) and (x < 530)
38      y_superior = (y >= 510) and (y < 530)
39      return (x_superior and y_superior)
40
41  def punto_3(x,y):
42      x_superior = (x >= 710) and (x < 810)
43      y_superior = (y >= 25) and (y < 60)
44      return (x_superior and y_superior)
```

Ilustración 35 Programación cuadricula para detección
Elaborado por: Melissa Rojas Yela

- Fila 11 y 12 se hace las respectivas importaciones de librerías usando la sentencia import “nombre de librería”.
- Fila 17 realiza una función de datetime para obtener la fecha del momento en el que se ejecuta el programa.

- Fila 18 define el nombre del archivo de texto plano (.txt) a crearse cuyo formato se deriva a partir de la fecha de ejecución del programa que sería AAAAMMDD.txt
- Fila 19 se crea dicho archivo con el nombre seleccionado y si ya hay un archivo de ese mismo nombre se procede a vaciar su contenido.
- Desde la fila 21 a la fila 24 se crea una función, una pequeña porción de código que ejecuta al ser llamada y la cual requiere parámetros de entrada (input) y obtenemos una salida (output), llamada cuadricula_superior que devolverá un valor booleano en el caso que se cumpla o no las condiciones.
- Se va a verificar en la fila 22 las condiciones que determina si el valor de x se encuentra entre 2 valores predefinidos.
- Fila 23 si el valor de y se encuentra entre 2 distintos valores predefinidos, va verificando.
- Fila 24 se devuelve al usuario si es que las condiciones se cumplieron.
- Desde la fila 26 hasta la hasta la fila 29 existe un similar procedimiento que la función descrita anteriormente lo que cambia son los valores predefinidos para las variables x y.
- De la fila 31 hasta la fila 34 ocurre un mismo procedimiento de comparación y cumplimientos de condiciones como en las funciones previamente descrito, pero como su nombre lo denota punto_5 hace hincapié a la localización del punto número 5 y se define un intervalo pequeño tanto para x como para y. Este procedimiento se verá repetido para los puntos 4, 3, 2 y 1.



```
45
46  def punto_1(x,y):
47      x_superior = (x >= 160) and (x < 260)
48      y_superior = (y >= 25) and (y < 60)
49      return (x_superior and y_superior)
50  def posicion(x,y):
51      if punto_1(x, y):
52          return 0
53      elif punto_3(x, y):
54          return 2
55      elif punto_4(x, y):
56          return 3
57      elif punto_5(x, y):
58          return 4
59      else:
60          return 1
61  def transformacion(frameId):
62      cantidad = int(frameId)
63      segundo = (cantidad // 50) % 60
64      minuto = cantidad // 3000
65      palabra = "{}:{}{}".format(minuto,'0' if segundo < 9 else '',segundo)
66      return palabra
67  flag = False
68  num_frame = 1
69  while cap.isOpened():
70    frameId = cap.get(1)
71    print("Frame numero "+str(num_frame))
72    num_frame += 1
73    ret,frame = cap.read()
74    ahora = datetime.datetime.now()
75    archivo_nombre = "{}.txt".format(ahora.strftime("%Y%m%d"))
76    elem_list = [0]*5
77    if (frameId % math.floor(frameRate) == 0):
78      flag = True
79      palabra = transformacion(frameId)
80      indice_frame = 1
```

Ilustración 36 Programación de condiciones de posiciones de puntos

Elaborado por: Melissa Rojas Yela

- Desde la fila 50 hasta la fila 60 se crea una función llamada posición que recibe como parámetros de entrada la coordenada y procede a dar un número entre 0 y 4. Esta función hace uso de las funciones creadas anteriormente.
- Fila 51 si cumple la condición que es el primer punto entonces en la fila 52 se procederá a dar el valor de 0, si no se cumple dicha condición se pasa a la fila 53 la cual pregunta si se cumple la condición del punto número 3 entonces pasa a la fila 54 para retornar el número 2.
- En caso contrario en la fila 55 se pregunta si cumple las condiciones para ser el punto número 4 y si llegara a cumplirla entonces pasa a la línea 56 a retornar un valor de 3, por otro lado, pasa a la fila 57 para preguntar si se cumple la condición de ser el punto número 5 y si llegara a cumplir pasa a la línea 58 a retornar el valor de 4, caso contrario la fila 59 denota que si no se llegó a cumplir todo lo anterior se pasa a la línea 60 a devolver un valor de 1.
- Desde la fila 61 hasta la fila 66 se crea la función transformación que recibe un identificador de un frame.
- Fila 62 se realiza una variable cantidad que es el valor numérico del identificador del frame.

- Fila 63 va obteniendo los segundos de acuerdo a la división de 50 y el residuo de 60.
- Fila 64 se obtiene el valor de los minutos al dividir la cantidad por 3000.
- Fila 65 se procede a crear el texto en el formato MM:SS con los segundos y minutos creados.
- Fila 66 retorna ese valor.
- Define en la fila 67 una variable booleana la cual permite escribir en el archivo de texto plano.
- Fila 68 se desarrolla una variable cuyo valor inicial es 1.
- Fila 76 se realiza una lista cuyos elementos son cincos 0.
- Fila 77 se pregunta si cumple las condiciones para ser un frame, en caso de cumplirla en la fila 78 la variable definida en la fila 67 cambia a True, en la fila 79 se obtiene el instante de tiempo en el video.
- Fila 80 obtiene el índice del frame.

```
84      maskRed = cv2.inRange(frameHSV, redBajo2, redAlto2)
85
86      contornos,_ = cv2.findContours(maskRed, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
87
88      for c in contornos:
89        area = cv2.contourArea(c)
90        if area > 100:
91          M = cv2.moments(c)
92          if (M["m00"]==0): M["m00"]=1
93          x = int(M["m10"]/M["m00"])
94          y = int(M['m01']/M['m00'])
95          if cuadricula_superior(x, y) or cuadricula_inferior(x, y):
96            cv2.circle(frame, (x,y), 5, (0,255,0), -1)
97            font = cv2.FONT_HERSHEY_SIMPLEX
98            cv2.putText(frame, '{},{}'.format(x,y),(x+10,y), font, 0.75,(0,255,0),1,cv2.LINE_AA)
99            if flag:
100                 elem_list[posicion(x, y)] = (x, y)
101             #print(dictionary)
102         nuevoContorno = cv2.convexHull(c)
103         cv2.drawContours(frame, [nuevoContorno], 0, (0,0,255), 3)
104     if flag:
105       with open(archivo_nombre, "a+") as myfile:
106         myfile.write(palabra+"-Frame-N°"+str(indice_frame) + " :["+";".join(str(v) for v in elem_list)+']\n')
107         myfile.close()
108       indice_frame += 1
109     cv2.imshow('frame', frame)
110     cv2.imshow('maskRed', maskRed)
111   #cv2.imshow('maskRedvis', maskRedvis)
112     if cv2.waitKey(30) & 0xFF == ord('s'):
113       break
114   else:
115     break
116
117
118 cap.release()
119 cv2.destroyAllWindows()
120
```

Ilustración 37 Programación para abrir un archivo con las lecturas

Elaborado por: Melissa Rojas Yela

- Fila 95 se pregunta si dicha coordenada se encuentra tanto en la cuadrícula superior y en la cuadrícula inferior.
- Fila 99 se pregunta si la variable booleana Flag es verdadera entonces pasa a ejecutar la fila 100, la cual se encarga de realizar la inserción

en la lista creada en la fila 76 de acuerdo a qué punto va a guardar haciendo uso de la función posición.

- Desde la fila 104 hasta la fila 108 se ejecuta en el caso que la variable booleana flag sea verdadera.
- Fila 105 se abre el archivo de texto plano a escribir y en la fila 106 se escribe una línea en el archivo de texto plano con los datos de las lecturas, el tiempo del video y el número de frame.
- Fila 107 se procede a cerrar el archivo a escribir por motivos de seguridad y en la fila 108 se aumenta el valor del índice del frame.

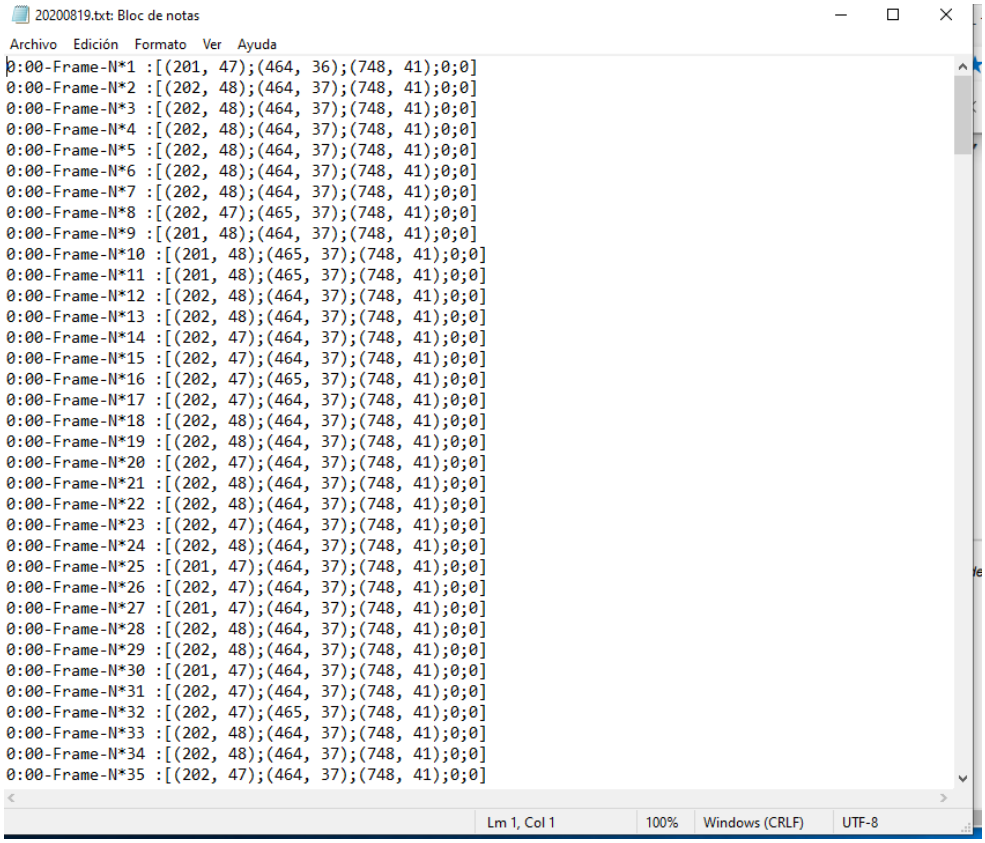
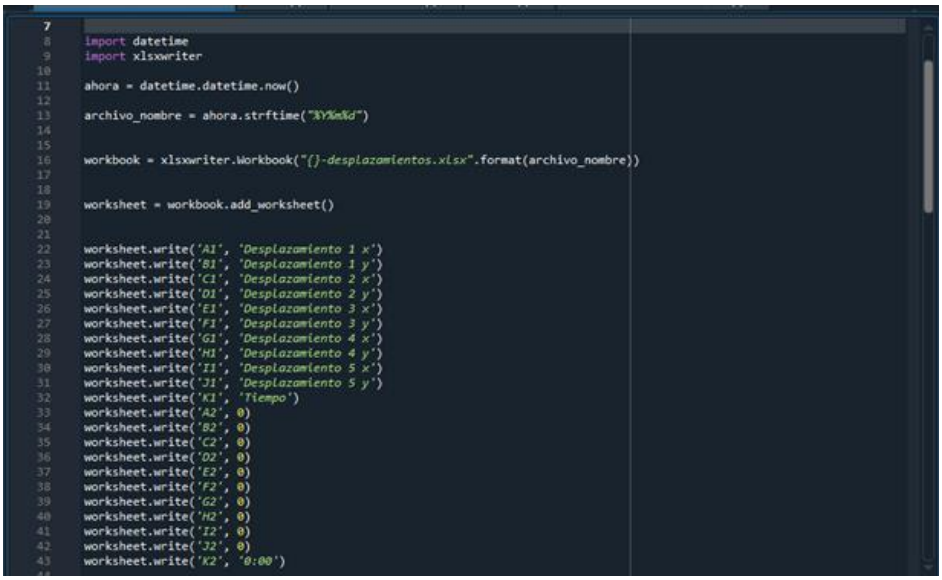


Ilustración 38 Respuesta de coordenadas de pixeles en bloc de notas

Elaborado por: Melissa Rojas Yela



```
import datetime
import xlsxwriter

ahora = datetime.datetime.now()

archivo_nombre = ahora.strftime("%Y%m%d")


workbook = xlsxwriter.Workbook("{}-desplazamientos.xlsx".format(archivo_nombre))


worksheet = workbook.add_worksheet()


worksheet.write('A1', 'Desplazamiento 1 x')
worksheet.write('B1', 'Desplazamiento 1 y')
worksheet.write('C1', 'Desplazamiento 2 x')
worksheet.write('D1', 'Desplazamiento 2 y')
worksheet.write('E1', 'Desplazamiento 3 x')
worksheet.write('F1', 'Desplazamiento 3 y')
worksheet.write('G1', 'Desplazamiento 4 x')
worksheet.write('H1', 'Desplazamiento 4 y')
worksheet.write('I1', 'Desplazamiento 5 x')
worksheet.write('J1', 'Desplazamiento 5 y')
worksheet.write('K1', 'Tiempo')
worksheet.write('A2', 0)
worksheet.write('B2', 0)
worksheet.write('C2', 0)
worksheet.write('D2', 0)
worksheet.write('E2', 0)
worksheet.write('F2', 0)
worksheet.write('G2', 0)
worksheet.write('H2', 0)
worksheet.write('I2', 0)
worksheet.write('J2', 0)
worksheet.write('K2', '0:00')
```

Ilustración 39 Programación para abrir archivo en Excel

Elaborado por: Melissa Rojas Yela

Se

crea un programa en Python que procesa los datos del archivo de texto plano creado y pasar dicho procesamiento a un archivo xlsx comúnmente llamado archivo EXCEL.

- Fila 8 se llama a la librería datetime para el manejo del tiempo y en la fila 9 se llama a la librería xlswriter para la escritura de archivos xlsx.
- Fila 11 se obtiene el tiempo actual y en la fila 13 del tiempo actual obtenido se convierte a un texto con formato AAAAMMDD.
- Fila 16 se procede a crear un Workbook con el nombre del archivo a crear y su respectiva hoja de cálculo en la fila 19.
- Desde la fila 22 hasta la fila 32 procede a realizar dicho encabezado denotando el valor a mostrar en cada columna y desde la fila 33 hasta la fila 43 se escriben en el archivo xlsx valores iniciales tanto 0 como 0:00.

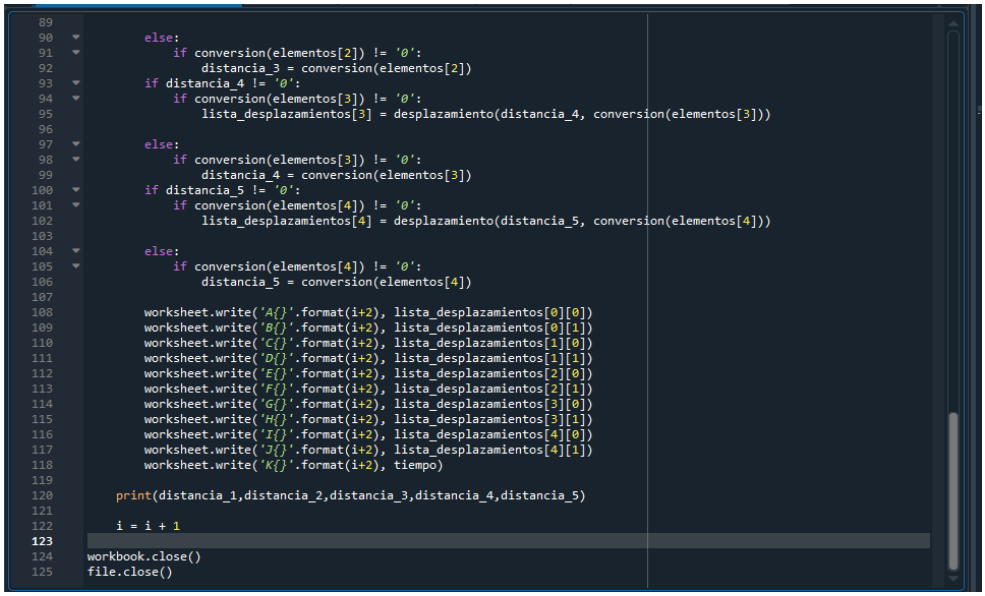
```
45  def conversion(elemento):
46      if elemento=='0':
47          return elemento
48      else:
49          elementos=elemento[1:-1].split(",")
50          return int(elementos[0]), int(elementos[1])
51
52  def desplazamiento(dist_1, dist_2):
53      x_inicial, y_inicial = dist_1
54      x_final, y_final = dist_2
55      return (x_final - x_inicial),(y_final - y_inicial)
56
57  file = open("{}.txt".format(archivo_nombre),"r")
58  i = 0
59  for line in file:
60      line = line.strip()
61      elementos = line[:-1].split('[')[1].split(';')
62      tiempo = line.split("-Frame")[0]
63      if i==0:
64          distancia_1 = conversion(elementos[0])
65          distancia_2 = conversion(elementos[1])
66          distancia_3 = conversion(elementos[2])
67          distancia_4 = conversion(elementos[3])
68          distancia_5 = conversion(elementos[4])
69      else:
70          lista_desplazamientos = [(0,0)]*5
71          if distancia_1 != '0':
72              if conversion(elementos[0]) != '0':
73                  lista_desplazamientos[0] = desplazamiento(distancia_1, conversion(elementos[0]))
74
75          else:
76              if conversion(elementos[0]) != '0':
77                  distancia_1 = conversion(elementos[0])
78          if distancia_2 != '0':
79              if conversion(elementos[1]) != '0':
80                  lista_desplazamientos[1] = desplazamiento(distancia_2, conversion(elementos[1]))
81
```

Ilustración 40 Programación de desplazamientos

Elaborado por: Melissa Rojas Yela

- Desde la fila 45 hasta la fila 50 se crea la función conversión en la que se procede a transformar los datos de texto en datos enteros. Esto se refiere que en el momento de lectura de un archivo externo al programa estos se encuentran en formato cadena de caracteres (string) pero es necesario usar datos numéricos (integer) en el cual se puedan realizar operaciones con dichos números.
- Desde la fila 52 hasta la fila 55 la función desplazamiento devuelve la diferencia entre 2 coordenadas, en este caso sería la coordenada final menos la coordenada inicial. Dado que las coordenadas tienen componentes x y entonces se procede a restar dichos componentes entre sí.
- En la fila 57 se procede a crear el archivo de texto plano en el cual están listadas las coordenadas xy de las lecturas del video.
- Fila 59 crea un lazo que empieza a iterar cada línea del archivo de texto plano.
- Fila 60 procede a quitar el salto de línea a la línea del archivo iterado.

- Fila 61 crea una lista de las coordenadas en dicha línea mientras que en la fila 62 se obtiene dicho tiempo del video que se encuentra el frame.
- Desde la fila 63 hasta la fila 68 si se cumple la condición que es la primera línea del archivo entonces se crean 5 distancias iniciales usando la función conversión y los elementos de la lista creada en la fila 61, si no llegara a cumplir dicha condición en la fila 70 se crea una lista de desplazamientos inicializados en coordenadas nulas (0,0). Dado que el archivo de texto plano puede presentar el carácter '0' y si no se cumple eso entonces se realiza la conversión del archivo de texto plano a una coordenada y se efectúa el desplazamiento entre dicha coordenada y la definida como coordenada inicial, esto está presente desde la fila 71 hasta la fila 77. Este procedimiento repite con los 5 puntos.



```
89
90          else:
91              if conversion(elementos[2]) != '0':
92                  distancia_3 = conversion(elementos[2])
93          if distancia_4 != '0':
94              if conversion(elementos[3]) != '0':
95                  lista_desplazamientos[3] = desplazamiento(distancia_4, conversion(elementos[3]))
96
97          else:
98              if conversion(elementos[3]) != '0':
99                  distancia_4 = conversion(elementos[3])
100         if distancia_5 != '0':
101             if conversion(elementos[4]) != '0':
102                 lista_desplazamientos[4] = desplazamiento(distancia_5, conversion(elementos[4]))
103
104         else:
105             if conversion(elementos[4]) != '0':
106                 distancia_5 = conversion(elementos[4])
107
108         worksheet.write('A{}'.format(i+2), lista_desplazamientos[0][0])
109         worksheet.write('B{}'.format(i+2), lista_desplazamientos[0][1])
110         worksheet.write('C{}'.format(i+2), lista_desplazamientos[1][0])
111         worksheet.write('D{}'.format(i+2), lista_desplazamientos[1][1])
112         worksheet.write('E{}'.format(i+2), lista_desplazamientos[2][0])
113         worksheet.write('F{}'.format(i+2), lista_desplazamientos[2][1])
114         worksheet.write('G{}'.format(i+2), lista_desplazamientos[3][0])
115         worksheet.write('H{}'.format(i+2), lista_desplazamientos[3][1])
116         worksheet.write('I{}'.format(i+2), lista_desplazamientos[4][0])
117         worksheet.write('J{}'.format(i+2), lista_desplazamientos[4][1])
118         worksheet.write('K{}'.format(i+2), tiempo)
119
120     print(distancia_1,distancia_2,distancia_3,distancia_4,distancia_5)
121
122     i = i + 1
123
124 workbook.close()
125 file.close()
```

Ilustración 41 Programación de posiciones de desplazamientos

Elaborado por: Melissa Rojas Yela

- Fila 108 hasta la fila 118 procede a escribir los desplazamientos tanto sus desplazamientos en x como en y al igual que el tiempo.
- Fila 120 se imprime por pantalla las distancias iniciales por control de datos.
- Fila 122 se aumenta el valor de la variable que ayuda a saber si se encuentra en la primera línea del archivo.

- Finalmente, tanto en la fila 124 y 125 se procede a cerrar el archivo xlsx y el archivo de texto plano respectivamente.

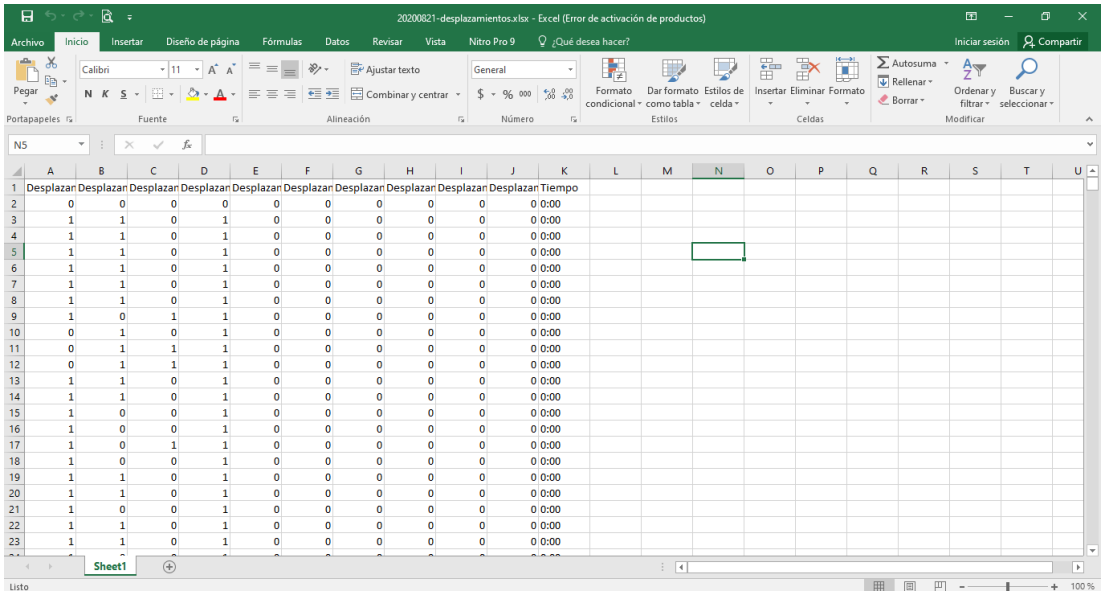


Ilustración 42 Respuesta de programación de desplazamientos

Elaborado por: Melissa Rojas Yela

Ya se observa lo que se tiene como resultados de desplazamiento de pixeles, tomando en cuenta que se debe multiplicar por el factor que haya sacado con el modelo real teniendo una relación entre las coordenadas y la distancia de los puntos.

Obteniendo las respuestas se procede a realizar las gráficas para una mejor visualización del contenido en sí.

Conclusión

Se ha realizado este manual en con fin que se pueda aprender lo que un programa aportar en la Ingeniería, y que la facultad de ingeniería pueda usar para hallar desplazamientos en diferentes ensayos por medios de videos que sean captados y procesados con detección de colores.

CAPITULO V

5 CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

Mediante las magnitudes diferentes de los dos ensayos dinámicos se puede notar como el programa ha tomado lectura alguna con una detección de un especifico color con la finalidad de tener respuesta de coordenadas de pixeles siendo leídas para sacar dichos desplazamientos.

Aplicar el procesamiento de imágenes es factible por el reconocimiento de imágenes, identificación de objetos, detección del color teniendo funcionalidades que los algoritmos permiten.

Las respuestas que se ha conseguido ha sido de gran similitud con respecto a los gráficos que se ha desarrollado con resultados de sensores y códigos para desplazamiento.

La importancia que tienen estos ensayos es tener el conocimiento de la respuesta de la estructura ante excitaciones dinámicas externas, como pueden ser sismos o vibraciones inducidas por maquinaria. Se pueden obtener diferentes mediciones, uno de ellos como desplazamiento.

En los ensayos dinámicos se nota claramente la diferencia que existe entre el programa con los sensores y que el uso del programa bien puede ser utilizado haciendo que siempre se realice con mayor precisión los enfoques óptimos de los puntos para obtener buenas lecturas de coordenadas de los pixeles y llegar a sacar el desplazamiento.

El realizar un manual de programación con Python resulta indispensable para cualquier uso que se le desea dar al programa siendo así que el uso que se le ha dado es sacar desplazamientos en ensayos mediante procesamiento de imágenes logrando una mayor eficiencia de los recursos que se puede llevar acabo en la facultad de ingeniería facilitando la estandarización de los métodos y del conocimiento que se adquiere.

5.2 Recomendaciones

Entre el espécimen y la cámara debe ser un metro para no perder especio de imagen ni resolución. Aclarando que el acercamiento óptico bien puede ser el acercar la cámara o realizando zoom desde la cámara con los diferentes lentes que puede tener una cámara analógica, el acercamiento digital es hacer zoom a una imagen fija para visualizar un objeto, pero esto permite perder la calidad. Lo que hacen la mayoría de dispositivos es acercamiento digital.

Por ende, se debe estar lejos del espécimen por seguridad utilizando cámaras con acercamiento óptico.

Se considere la propuesta del manual para que en la facultad de ingeniería se tenga una nueva herramienta para la ejecución de los ensayos dinámicos y estáticos.

Asimismo, que se proporcione a los estudiantes este manual que los beneficiará facilitando el aprendizaje de la programación que se ha realizado.

Que este manual sea actualizado mejorándolo constantemente, con los cambios o ediciones que sean necesarias tanto en la programación como en la ejecución de los ensayos, tomando mucho en cuenta que al desarrollarse tanto el ensayo como la filmación se tiene que ser precisos con el color y la ubicación del enfoque del instrumento que se use para realizar el video o imagen.

ANEXOS

Se presenta todo lo generado con Python a continuación para la obtención de coordenadas de pixeles.

*DETECCIÓN DE VIDEOS-DINAMICO.py: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
@author: Melissa
"""


import cv2
import numpy as np
import datetime
import math

cap = cv2.VideoCapture("E3F1sup.mp4")
frameRate = cap.get(5)

redBajo2=np.array([175, 100, 20], np.uint8)
redAlto2=np.array([180, 255, 255], np.uint8)

ahora = datetime.datetime.now()
archivo_nombre = "{}.txt".format(ahora.strftime("%Y%m%d"))
open(archivo_nombre,"w+").close()

def cuadricula_superior(x,y):
    x_superior = (x >= 160) and (x < 810)
    y_superior = (y >= 25) and (y < 60)
    return (x_superior and y_superior)

def cuadricula_inferior(x,y):
    x_superior = (x >= 500) and (x < 550)
    y_superior = (y >= 400) and (y < 550)
    return (x_superior and y_superior)

def punto_5(x,y):
    x_superior = (x >= 500) and (x < 550)
    y_superior = (y >= 400) and (y < 550)
    return (x_superior and y_superior)
```

Ln 38, Col 1 | 100% | Windows (CRLF) | UTF-8

*DETECCIÓN DE VIDEOS-DINAMICO.py: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
def punto_4(x,y):
    x_superior = (x >= 440) and (x < 530)
    y_superior = (y >= 510) and (y < 530)
    return (x_superior and y_superior)

def punto_3(x,y):
    x_superior = (x >= 710) and (x < 810)
    y_superior = (y >= 25) and (y < 60)
    return (x_superior and y_superior)

def punto_1(x,y):
    x_superior = (x >= 160) and (x < 260)
    y_superior = (y >= 25) and (y < 60)
    return (x_superior and y_superior)
def posicion(x,y):
     if punto_1(x, y):
         return 0
     elif punto_3(x, y):
         return 2
     elif punto_4(x, y):
         return 3
     elif punto_5(x, y):
         return 4
     else:
         return 1
def transformacion(frameId):
    cantidad = int(frameId)
    segundo = (cantidad // 50) % 60
    minuto = cantidad // 3000
    palabra = "{}:{}{}".format(minuto,'0' if segundo < 9 else '',segundo)
    return palabra
flag = False
num_frame = 1
while cap.isOpened():
```

Ln 72, Col 1 | 100% | Windows (CRLF) | UTF-8

*DETECCIÓN DE VIDEOS-DINAMICO.py: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
def transformacion(frameId):
    cantidad = int(frameId)
    segundo = (cantidad // 50) % 60
    minuto = cantidad // 3000
    palabra = "{}:{}{}".format(minuto,'0' if segundo < 9 else '',segundo)
    return palabra
flag = False
num_frame = 1
while cap.isOpened():
  frameId = cap.get(1)
  print("Frame numero "+str(num_frame))
  num_frame += 1
  ret,frame = cap.read()
  ahora = datetime.datetime.now()
  archivo_nombre = "{}.txt".format(ahora.strftime("%Y%m%d"))
  elem_list = [0]*5
  if (frameId % math.floor(frameRate) == 0):
    flag = True
    palabra = transformacion(frameId)
    indice_frame = 1
  if ret:
    frameHSV = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    maskRed = cv2.inRange(frameHSV, redBajo2, redAlto2)
    contornos,_ = cv2.findContours(maskRed, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    for c in contornos:
      area = cv2.contourArea(c)
      if area > 100:
        M = cv2.moments(c)
        if (M["m00"]==0): M["m00"]=1
        x = int(M["m10"]/M["m00"])
        y = int(M['m01']/M['m00'])
        if cuadricula_superior(x, y) or cuadricula_inferior(x, y):
          cv2.circle(frame, (x,y), 5, (0,255,0), -1)
          font = cv2.FONT_HERSHEY_SIMPLEX
          cv2.putText(frame, '{},{}'.format(x,y),(x+10,y), font, 0.75,(0,255,0),1,cv2.LINE_AA)
```

Lm 100, Col 53 | 100% | Windows (CRLF) | UTF-8

*DETECCIÓN DE VIDEOS-DINAMICO.py: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
    maskRed = cv2.inRange(frameHSV, redBajo2, redAlto2)
    contornos,_ = cv2.findContours(maskRed, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    for c in contornos:
      area = cv2.contourArea(c)
      if area > 100:
        M = cv2.moments(c)
        if (M["m00"]==0): M["m00"]=1
        x = int(M["m10"]/M["m00"])
        y = int(M['m01']/M['m00'])
        if cuadricula_superior(x, y) or cuadricula_inferior(x, y):
          cv2.circle(frame, (x,y), 5, (0,255,0), -1)
          font = cv2.FONT_HERSHEY_SIMPLEX
          cv2.putText(frame, '{},{}'.format(x,y),(x+10,y), font, 0.75,(0,255,0),1,cv2.LINE_AA)
          if flag:
                elem_list[posicion(x, y)] = (x, y)
        nuevoContorno = cv2.convexHull(c)
        cv2.drawContours(frame, [nuevoContorno], 0, (0,0,255), 3)
    if flag:
      with open(archivo_nombre, "a+") as myfile:
        myfile.write(palabra+"-Frame-N*"+str(indice_frame) + " :["+";".join(str(v) for v in elem_list)
        myfile.close()
      indice_frame += 1
    cv2.imshow('frame', frame)
    cv2.imshow('maskRed', maskRed)

    if cv2.waitKey(30) & 0xFF == ord('s'):
      break
  else:
    break


cap.release()
cv2.destroyAllWindows()
```

Lm 120, Col 1 | 100% | Windows (CRLF) | UTF-8

Codigos para obtener resultados de desplazamientos en excel

*procesamiento-desplazamientos.py: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
@author: Melissa
"""
import datetime
import xlsxwriter

ahora = datetime.datetime.now()
archivo_nombre = ahora.strftime("%Y%m%d")
workbook = xlsxwriter.Workbook("{}-desplazamientos.xlsx".format(archivo_nombre))
worksheet = workbook.add_worksheet()

worksheet.write('A1', 'Desplazamiento 1 x')
worksheet.write('B1', 'Desplazamiento 1 y')
worksheet.write('C1', 'Desplazamiento 2 x')
worksheet.write('D1', 'Desplazamiento 2 y')
worksheet.write('E1', 'Desplazamiento 3 x')
worksheet.write('F1', 'Desplazamiento 3 y')
worksheet.write('G1', 'Desplazamiento 4 x')
worksheet.write('H1', 'Desplazamiento 4 y')
worksheet.write('I1', 'Desplazamiento 5 x')
worksheet.write('J1', 'Desplazamiento 5 y')
worksheet.write('K1', 'Tiempo')
worksheet.write('A2', 0)
worksheet.write('B2', 0)
worksheet.write('C2', 0)
worksheet.write('D2', 0)
worksheet.write('E2', 0)
worksheet.write('F2', 0)
worksheet.write('G2', 0)
worksheet.write('H2', 0)
worksheet.write('I2', 0)
worksheet.write('J2', 0)
worksheet.write('K2', '0:00')
```

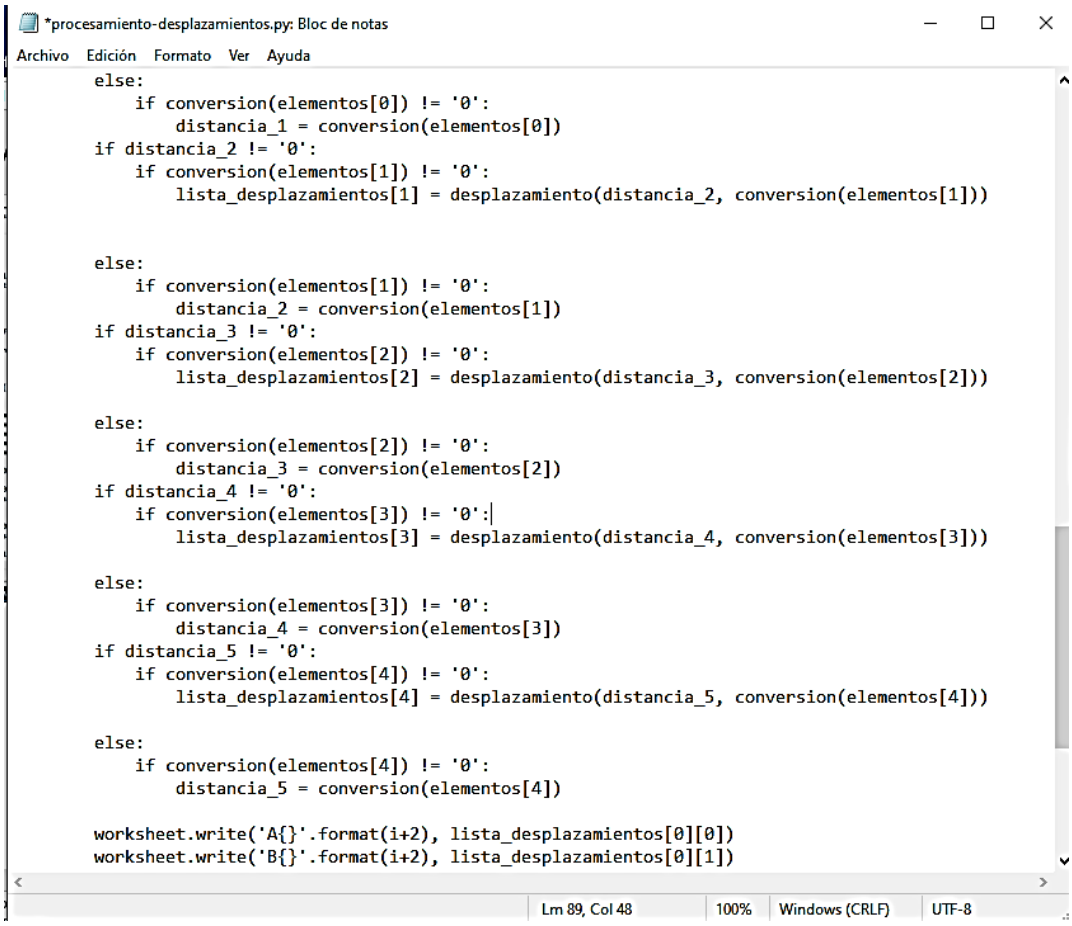
Ln 39, Col 1 100% Windows (CRLF) UTF-8

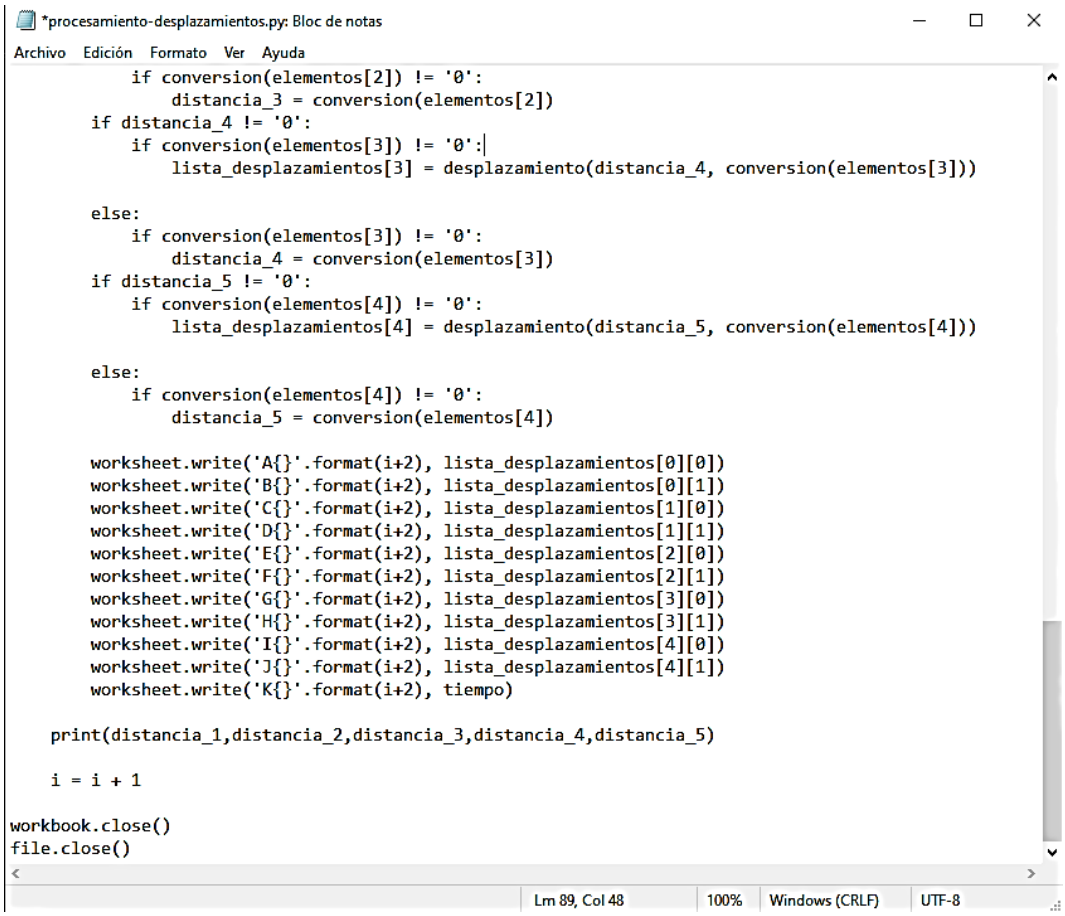
*procesamiento-desplazamientos.py: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
def conversion(elemento):
    if elemento=='0':
        return elemento
    else:
        elementos=elemento[1:-1].split(",")
        return int(elementos[0]), int(elementos[1])

def desplazamiento(dist_1, dist_2):
    x_inicial, y_inicial = dist_1
    x_final, y_final = dist_2
    return (x_final - x_inicial),(y_final - y_inicial)

file = open("{}.txt".format(archivo_nombre),"r")
i = 0
for line in file:
    line = line.strip()
    elementos = line[:-1].split('[')[1].split(';')
    tiempo = line.split("-Frame")[0]
    if i==0:
        distancia_1 = conversion(elementos[0])
        distancia_2 = conversion(elementos[1])
        distancia_3 = conversion(elementos[2])
        distancia_4 = conversion(elementos[3])
        distancia_5 = conversion(elementos[4])
    else:
        lista_desplazamientos = [(0,0)]*5
        if distancia_1 != '0':
            if conversion(elementos[0]) != '0':
                lista_desplazamientos[0] = desplazamiento(distancia_1, conversion(elementos[0]))

        else:
            if conversion(elementos[0]) != '0':
                distancia_1 = conversion(elementos[0])
        if distancia_2 != '0':
            if conversion(elementos[1]) != '0':
```

Ln 24, Col 44 100% Windows (CRLF) UTF-8


*procesamiento-desplazamientos.py: Bloc de notas
Archivo Edición Formato Ver Ayuda
else:
if conversion(elementos[0]) != '0':
distancia_1 = conversion(elementos[0])
if distancia_2 != '0':
if conversion(elementos[1]) != '0':
lista_desplazamientos[1] = desplazamiento(distancia_2, conversion(elementos[1]))
else:
if conversion(elementos[1]) != '0':
distancia_2 = conversion(elementos[1])
if distancia_3 != '0':
if conversion(elementos[2]) != '0':
lista_desplazamientos[2] = desplazamiento(distancia_3, conversion(elementos[2]))
else:
if conversion(elementos[2]) != '0':
distancia_3 = conversion(elementos[2])
if distancia_4 != '0':
if conversion(elementos[3]) != '0':
lista_desplazamientos[3] = desplazamiento(distancia_4, conversion(elementos[3]))
else:
if conversion(elementos[3]) != '0':
distancia_4 = conversion(elementos[3])
if distancia_5 != '0':
if conversion(elementos[4]) != '0':
lista_desplazamientos[4] = desplazamiento(distancia_5, conversion(elementos[4]))
else:
if conversion(elementos[4]) != '0':
distancia_5 = conversion(elementos[4])
worksheet.write('A{}'.format(i+2), lista_desplazamientos[0][0])
worksheet.write('B{}'.format(i+2), lista_desplazamientos[0][1])
Lm 89, Col 48
100%
Windows (CRLF)
UTF-8


*procesamiento-desplazamientos.py: Bloc de notas
Archivo Edición Formato Ver Ayuda
if conversion(elementos[2]) != '0':
distancia_3 = conversion(elementos[2])
if distancia_4 != '0':
if conversion(elementos[3]) != '0':
lista_desplazamientos[3] = desplazamiento(distancia_4, conversion(elementos[3]))
else:
if conversion(elementos[3]) != '0':
distancia_4 = conversion(elementos[3])
if distancia_5 != '0':
if conversion(elementos[4]) != '0':
lista_desplazamientos[4] = desplazamiento(distancia_5, conversion(elementos[4]))
else:
if conversion(elementos[4]) != '0':
distancia_5 = conversion(elementos[4])
worksheet.write('A{}'.format(i+2), lista_desplazamientos[0][0])
worksheet.write('B{}'.format(i+2), lista_desplazamientos[0][1])
worksheet.write('C{}'.format(i+2), lista_desplazamientos[1][0])
worksheet.write('D{}'.format(i+2), lista_desplazamientos[1][1])
worksheet.write('E{}'.format(i+2), lista_desplazamientos[2][0])
worksheet.write('F{}'.format(i+2), lista_desplazamientos[2][1])
worksheet.write('G{}'.format(i+2), lista_desplazamientos[3][0])
worksheet.write('H{}'.format(i+2), lista_desplazamientos[3][1])
worksheet.write('I{}'.format(i+2), lista_desplazamientos[4][0])
worksheet.write('J{}'.format(i+2), lista_desplazamientos[4][1])
worksheet.write('K{}'.format(i+2), tiempo)
print(distancia_1,distancia_2,distancia_3,distancia_4,distancia_5)
i = i + 1
workbook.close()
file.close()
Lm 89, Col 48
100%
Windows (CRLF)
UTF-8

REFERENCIAS

Bertero, A., & Bertero, R. (2014). ENSAYOS DINÁMICOS SOBRE UNA COLUMNA DE HORMIGÓN ARMADO A ESCALA EN MESA VIBRADORA. *Ensayos Dinámicos Sobre Una Columna de Hormigón Armado a Escala En Mesa Vibradora*, *1*(1), 1–20.

Challenger Pérez, I., Díaz-Ricardo, Y., & Becerra -García, R. A. (2014). El lenguaje de programación Python. *Ciencias Holguín*, *XX*(Industria Informática), 1–13. http://www.linuxjournal.com/article/2959

Coelho, L. P. (2013). Formats Supported. *Imread 0.6*, *1*(Software de código abierto para la visión por computadora con guiones), 1. https://pythonhosted.org/imread/formats.html

Gómez, D., & Guerrero, A. (2016). *ESTUDIO Y ANÁLISIS DE TÉCNICAS PARA PROCESAMIENTO DIGITAL DE IMÁGENES.* [Universidad Tecnológica de Pereira]. http://repositorio.utp.edu.co/dspace/bitstream/handle/11059/6494/00642G633.pdf?sequenc

González, L. (2018). *Introducción al IDE Spyder*. Aprende Todo Sobre Inteligencia Superficial. https://ligdigonzalez.com/ide-spyder-para-python/

Infotecnology. (2019). Sistemas operativos de código abierto: ¿cuál es la mejor opción? *Infotecnology*. https://www.infotechnology.com/online/Sistemas-operativos-de-codigo-abierto-cual-es-la-mejor-opcion-20191220-0008.html

Instron. (SA). (2020, August). *Ensayo de flexión* . INSTRON SOUTH AMERICA. https://www.instron.com.ar/es-ar/our-company/library/glossary/f/flexure-test

Marín, R. (2020, February 12). *OpenCV, Instalación en Python y ejemplos básicos*. Informática y Tics. https://revistadigital.inesem.es/informatica-y-tics/opencv/

Marzal, A., Gracía, I., & García, P. (2014). *Introducción a la programación con Python 3* (Primera). Universitat Jaume. https://doi.org/10.6035/Sapientia93

Ortego, D. (2017, June 6). *Los 5 mejores editores Python*. OpenWebinars. https://openwebinars.net/blog/los-5-mejores-editores-python/

Pérez, P., & Valente, M. (2018). *Fundamentos básicos del procesamiento de imágenes*. FAMAF, Facultad de Matemáticas, Astronomía, Física y Computación de La Universidad de Córdova-Argentina. https://www.famaf.unc.edu.ar/~pperez1/manuales/cim/cap2.html

Robert Johansson. (2016). *Introduction to Scientific Computing in Python*.

Rodríguez, O., Hernández, A., & Huerta, J. (2014). *Sistema de medición de distancia mediante imágenes para determinar la posición de una esfera utilizando el sensor Kinect XBOX*. http://www.scielo.org.mx/pdf/poli/n49/n49a8.pdf

Solano, G. (2019a). *Detección de colores en OpenCV - Python (En 4 pasos)*. OMES. https://omes-va.com/deteccion-de-colores/

Solano, G. (2019b, September 20). *DETECCIÓN DE COLORES Y Tracking en OpenCV*. OMES. https://omes-va.com/deteccion-de-colores2/

DECLARACIÓN Y AUTORIZACIÓN

Yo, **Rojas Yela, Melissa Michelle,** con C.C: # **1205404641** autora del trabajo de titulación: **Presentación y desarrollo de alternativa de medición de desplazamientos en ensayos de laboratorio empleando técnicas de procesamiento de imágenes para la Facultad de Ingeniería** previo a la obtención del título de **Ingeniería Civil** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, **17** de **septiembre** del **2020**

f. _____

Nombre: **Rojas Yela, Melissa Michelle**

C.C: **1205404641**

Presidencia de la República del Ecuador

Plan Nacional de Ciencia, Tecnología, Innovación y Saberes

SENESCYT
Secretaría Nacional de Educación Superior, Ciencia, Tecnología e Innovación

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA

FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN

TEMA Y SUBTEMA:	Presentación y desarrollo de alternativa de medición de desplazamientos en ensayos de laboratorio empleando técnicas de procesamiento de imágenes para la Facultad de Ingeniería		
AUTOR(ES)	Melissa Michelle, Rojas Yela		
REVISOR(ES)/TUTOR(ES)	Ing. Guillermo Ponce, M.Sc		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Ingeniería		
CARRERA:	Ingeniería Civil		
TITULO OBTENIDO:	Ingeniero Civil		
FECHA DE PUBLICACIÓN:	17 de septiembre de 2020	No. DE PÁGINAS:	77
ÁREAS TEMÁTICAS:	Programación, Métodos Numéricos, Estructuras		
PALABRAS CLAVES/ KEYWORDS:	Mediciones, Desplazamiento, Intérpretes, Editores, Código Abierto, Software Libre, Programación, Python		

RESUMEN/ABSTRACT:

Este documento es producto de una investigación realizada bibliográficamente y en el campo de la experimentación, cuyo objetivo motivador para la realización de este proyecto es presentar, desarrollar e implementar una metodología práctica para realizar las mediciones de desplazamientos a partir del procesamiento de imágenes tomados de ensayos de laboratorio, de tal modo que la Facultad de Ingeniería tenga una nueva y versátil herramienta para optimizar las condiciones de enseñanza de y mejorar el comportamiento de materiales y capacidad de los elementos. Para el desarrollo del proyecto se ha recurrido en la utilización del lenguaje de programación interpretador Python mediante su programa Editor Spyder que es con el que mejor se desarrolla, asimismo la utilización de la gran biblioteca OpenCv que funciona óptimamente con Python. El análisis y medición de los desplazamientos se hará con imágenes tomadas de videos tomadas de ensayos realizados experimentalmente de estructuras construidas a propósito para la ejecución del proyecto. Se ha tenido muy en cuenta obviamente la ventaja de la característica de ser programas de código abierto tanto de Python cuanto los editores y bibliotecas, se ha considerado puntualmente las aplicaciones técnicas para el análisis de las imágenes y sus colores, considerando también las escalas de grises y la disposición matricial y vectorial de los mismos con su resolución en pixeles, tamaño y combinación de colores.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** +593-989923221	E-mail: **melissa_rojas_@hotmail.com**
CONTACTO CON LA INSTITUCIÓN (C00RDINADOR DEL PROCESO UTE)::	**Nombre: Clara Glas Cevallos**	
	Teléfono: +593-4 -2206956	
	E-mail: clara.glas@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA	
Nº. DE REGISTRO (en base a datos):	
Nº. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	