UNIVERSIDAD NACIONAL DE CAAGUAZÚ

FACULTAD DE CIENCIAS Y TECNOLOGAS

INGENIERÍA CIVIL

“MÉTODO DE INTERPRETACIÓN AUTOMATIZADA DE BOCETOS ARQUITECTÓNICOS PARA LA OBTENCIÓN DE PLANOS EN PLANTA CAD, CÓMPUTOS DE LOS DIFERENTES ELEMENTOS CONSTRUCTIVOS, PREDIMENSIONAMIENTO Y SOLICITACIONES DE LOSAS.”

Elaborado por:

MARCOS VIRGILIO FLORES BARRETO

KEVIN ALEXANDER FERREIRA NUBERGER

Tutor:

Prof. Msc. Ing. Fredy Gabriel Ramírez Villanueva

Coronel Oviedo - Paraguay

Año 2022

PÁGINA DE APROBACIÓN

Trabajo de fin de grado para la obtención del Título de Ingeniera Civil aprobado en representación de la Facultad de Ciencias y Tecnologías de la Universidad Nacional de Caaguazú, por el tribunal Examinador constituido por los siguientes profesores.

_____

Prof. Ing.

_____

Prof. Ing.

_____

Prof. Ing.

Acta Nº:_____

Fecha: _____

Calificación Obtenida: _____

A nuestras familias.

AGRADECIMIENTOS:

- A Dios, por la vida y darnos la fortaleza para seguir adelante.
- A nuestras familias, por brindarnos la oportunidad de estudiar y acompañarnos en todos estos años.
- A la Facultad de Ciencias y Tecnologías de la Universidad Nacional de Caaguazú, por tratar de ofrecer los mejores profesores para tener una buena educación.
- A nuestro tutor, el Ingeniero Fredy Ramírez, por la buena predisposición y orientación que nos brindó para poder encaminar este trabajo.
- A todos aquellos profesores que nos brindaron sus conocimientos y que nos inculcaron a seguir aprendiendo.

RESUMEN

En el presente trabajo, se utilizó el sistema de detección de objetos basado en redes neuronales YOLO (You Only Look Once). El mismo se entrenó para detectar los elementos componentes de un boceto arquitectónico hecho a mano definidas por las siguientes clases; pared, puerta, ventana, cota, números del 0 al 9, coma, losa, sanitario.

Se desarrolló un algoritmo, el cual toma de base a las detecciones y a partir de los objetos que forman los números, agrupa los dígitos individuales en conjuntos de números decimales que representan a las medidas de las cotas en el espacio real de dibujo, estos valores se asignan a las cotas que corresponden y se crean los ejes que contendrán a las mamposterías. Seguidamente el algoritmo ubica las puertas y ventanas en el lugar donde deberían estar, como también toma a las losas detectadas y las asigna a escala. Los valores que se obtienen en este algoritmo son almacenados en matrices que definen a los elementos mencionados. Dichas matrices son clave a la hora de obtener los resultados buscados, ya que son los datos de entrada para los demás algoritmos.

Se desarrollaron tres algoritmos que a partir de las matrices obtenidas en el algoritmo de reconstrucción, cada uno tiene una función distinta, las cuales son: crear planos en planta CAD; hallar cómputos de diversos elementos constructivos; predimensionar y calcular solicitaciones de las losas.

ABSTRACT

In the present work, the object detection system based on neural networks YOLO (You Only Look Once) was used. It was trained to detect the component elements of a handmade architectural sketch defined by the following classes; pared, puerta, ventana, cota, numbers from 0 to 9, coma, losa, sanitario.

An algorithm was developed, which is based on the detections and from the objects that form the numbers, groups the individual digits in sets of decimal numbers that represent the measurements of the dimensions in the real drawing space, these values are assigned to the corresponding heights and the axes that will contain the masonry are created. Next, the algorithm places the doors and windows in the place where they should be, as well as takes the detected slabs and assigns them to scale. The values obtained in this algorithm are stored in arrays that define the mentioned elements. These arrays are key when it comes to obtaining the desired results, since they are the input data for the other algorithms.

Three algorithms were developed that from the matrices obtained in the reconstruction algorithm, each one has a different function, which are: create CAD plan plans; find counts of various construction elements; Pre-dimension and calculate requests for the slabs.

ÍNDICE

Capítulo 1 1
Introducción 1
1.1 Planteamiento del problema 2
1.2 Solución propuesta 3
1.3 Justificación 3
1.4 Objetivos 5
1.4.1 Objetivo General 5
1.4.2 Objetivos específicos 5
1.5 Antecedentes 6
1.6 Metodología 7
1.6.1 Creación de Base de datos 8
1.6.2 Entrenamiento del detector y prueba 8
1.6.3 Elaboración de algoritmo para corrección de coordenadas 8
1.6.4 Elaboración de algoritmos para creación de planos, cómputos y cálculo de losas 9
1.6.5 Normas 9
1.7 Alcance 9
Capítulo 2 11
Conceptos Preliminares 11
2.1 Aprendizaje Automático 11
2.2 Redes Neuronales 12

2.2.1 Arquitectura del perceptrón 13
2.2.2 Función de Activación 14
2.2.3 Proceso de aprendizaje del Perceptrón Multicapa 18
2.3 Detección de objetos 21
2.3.1 Redes neuronales convolucionales 21
2.3.2 Entrenamiento 24
2.3.2.1 Creación de Base de datos 24
2.3.2.2 Configuraciones previas 25
2.3.2.3 Proceso de Detección 26
2.4 Python 27
2.5 YOLO 28
2.5.1 Funcionamiento de YOLO 28
2.5.2 Arquitectura 31
2.5.3 Evolución de YOLO 33
2.5.3.1 YOLO v2 33
2.5.3.2 YOLO v3 34
2.5.3.3 YOLO v4 36
2.6 Losas 38
2.6.1 Tipos de losas según su dirección de armado 38
2.6.2 Tipos de losas según sus apoyos 39

2.6.3 Condiciones de apoyo 43
2.6.4 Tipos de cargas 44
2.6.4.1 Carga Muerta 44
2.6.4.2 Carga viva 44
2.6.4.3 Combinaciones de carga 44
2.6.5 Especificaciones para predimensionamiento y cálculo de solicitaciones 45
2.6.6 Cálculo de solicitaciones para losas macizas unidireccionales 45
2.6.7 Calculo de solicitaciones para losas macizas bidireccionales 46
2.7 Cómputo Métrico 46
2.7.1 Tareas preliminares 47
2.7.2 Movimiento de suelo 47
2.7.3 Estructura 47
2.7.4 Mamposterías 48
2.7.5 Revoques 48
2.7.6 Techos 48
2.7.7 Contrapisos 48
2.7.8 Pisos 49
2.7.9 Revestimientos 49
2.7.10 Vidrios y Cristales 49

2.7.11 Pintura y Empapelado 49

Capítulo 3 50
Funcionamiento del método de interpretación de bocetos 50
3.1 Configuraciones previas 51
3.2 Creación de Base de datos 52
3.2.1 Creación de habitaciones 55
3.2.2 Creación de puertas y ventanas 56
3.2.3 Creación de cotas y sus valores 56
3.3 Entrenamiento 57
3.4 Algoritmo Ing. Bob 60
3.5 Algoritmo de Reconstrucción 61
3.5.1 Interpretación de los números 61
3.5.2 Interpretación de las cotas 66
3.5.3 Ubicación de los ejes 68
3.5.4 Ubicación de mamposterías 74
3.5.5 Escala real de las mamposterías 76
3.5.6 Ubicación de aberturas 77
3.5.6.2 Ubicación de ventanas 82
3.5.7 Asignación de losas a escala 85
3.6 Algoritmo para obtención de planos CAD 90
3.6.1 Trazado de mamposterías y aberturas de puertas 91

3.6.2 Trazado de puertas 92
3.6.3 Trazado de ventanas 93
3.6.4 Cotas 93
3.7 Algoritmo para obtención de Cómputos 94
3.7.1 Cálculo de perímetros 95
3.7.2 Cálculo de áreas 95
3.7.3 Asignación de baños 97
3.7.4 Longitud y sección transversal de mamposterías 97
3.7.5 Cantidad de puertas 98
3.7.6 Área de mamposterías 98
3.7.7 Área de pisos 98
3.7.8 Áreas de mamposterías externas e internas 98
3.7.9 Área de azulejos 99
3.7.10 Volumen de hormigón 99
3.7.11 Exportación de resultados a planilla de Excel 100
3.8 Algoritmo de Cálculo de Solicitaciones en Losas 102
3.8.1 Datos de entrada 103
3.8.2 Determinación de dirección de armado 104
3.8.3 Predimensionamiento de espesor 105
3.8.4 Condiciones de apoyo 105

3.8.5 Análisis de cargas 106

3.8.6 Cálculo de Momentos y Cortantes 107

3.8.7 Gráficos 110

3.8.8 Ejemplos de prueba 110

Capítulo 4 117
Ejemplo de Uso 117
Capítulo 5 130
Análisis de costos 130
5.1 Análisis de costos de planos 130

5.2 Análisis de costos de cómputos 131

5.3 Análisis de costos de cálculo estructural de losas 131

Capítulo 6 133
Conclusiones 133
6.1 Recomendaciones y trabajos futuros 134

Apéndices 135
A.1 Cargas permanentes 136

A.2 Cargas variables 138

A.3 Método ACI 139

A.4 Tablas de Marcus 140

Referencias Bibliográficas 144

íNDICE DE FIGURAS

Figura 1 13
Arquitectura de Perceptrón..... 13

Figura 2 14
Función Sigmoidea..... 14

Figura 3 15
Función Hiperbólica..... 15

Figura 4 16
Función ReLU 16

Figura 5 16
Función Leaky ReLU..... 16

Figura 6 17
Arquitectura de Perceptrón multicapa 17

Figura 7 21
Imagen de una naranja (9 pixeles por 9 pixeles).. 21

Figura 8 22
Representación de Colores RGB en Espacio Tridimensional 22

Figura 9 23
Ejemplo de Filtro para Detectar Bordes..... 23

Figura 10 24
Arquitectura de Red Neuronal Convolucional..... 24

Figura 11 29
Detección realizada por YOLO.. 29

Figura 12 30
Intersección sobre la unión..... 30

Figura 13 31
Sistema de Detección de YOLO. 31

Figura 14 32
Arquitectura de YOLO. 32

Figura 15 39
Losa Armada en una Dirección. 39

Figura 16 39
Losa Armada en dos Direcciones. 39

Figura 17 40
Losa Apoyada en sus cuatro lados. 40

Figura 18 40
Losa con lado menor empotrado. 40

Figura 19 40
Losa con lado mayor empotrado. 40

Figura 20 41
Losa con lados menores empotrados. 41

Figura 21 41
Losa con lados mayores empotrados. 41

Figura 22 41
Losa con lados perpendiculares empotrados. 41

Figura 23 42
Losa con lados mayores y un lado menor empotrado. 42

Figura 24 42
Losa con lados menores y un lado mayor empotrado. 42

Figura 25 42

Losa con cuatro lados empotrados. 42

Figura 26 52
Modificación del archivo image.c. 52

Figura 27 54
Diagrama de flujo de Algoritmo para obtención de bocetos arquitectónicos. 54

Figura 28 57
Línea de código que importa Colab a Drive 57

Figura 29 58
Ventana de autorización para acceso a Drive. 58

Figura 30 59
Gráfico de porcentajes de aciertos y pérdida del entrenamiento. 59

Figura 31 60
Diagrama de flujo de Algoritmo Ing. Bob. 60

Figura 32 90
Diagrama de flujo de Algoritmo para obtención de planos CAD. 90

Figura 33 94
Diagrama de flujo de Algoritmo de Cómputos. 94

Figura 34 102
Diagrama de flujo de Algoritmo de Losas. 102

Figura 35 110
Ejemplo de losas unidireccionales. 110

Figura 36 111
Solución del ejemplo de losas unidireccionales. 111

Figura 37 111
Datos de partida 111

Figura 38 112
Espesores obtenidos del algoritmo de losas. 112

Figura 39 112
Carga permanente mayorada. 112

Figura 40 113
Carga variable mayorada. 113

Figura 41 113
Valores de Momentos y Cortantes. 113

Figura 42 113
Ejemplo de losas bidireccionales. 113

Figura 43 114
Solución del ejemplo de losas bidireccionales. 114

Figura 44 115
Datos de partida. 115

Figura 45 115
Espesores obtenidos del algoritmo de losas. 115

Figura 46 116
Carga permanente mayorada. 116

Figura 47 116
Carga variable mayorada. 116

Figura 48 116
Valores de Momentos y Cortantes. 116

Figura 49 117

Captura de pantalla de Ing. Bob Telegram. 117

Figura 50 118
Detecciones realizadas por Ing. Bob Telegram. 118

Figura 51 119
Captura de pantalla de edición en LabelImg Master 1. 119

Figura 52 119
Captura de pantalla de edición en LabelImg Master 2. 119

Figura 53 120
Captura de pantalla de edición en LabelImg Master 3. 120

Figura 54 120
Captura de pantalla de proceso de creación de cotas. 120

Figura 55 121
Captura de pantalla de proceso de creación de mamposterías 1. 121

Figura 56 121
Captura de pantalla de proceso de creación de mamposterías 2. 121

Figura 57 122
Captura de pantalla de proceso de creación de mamposterías 3. 122

Figura 58 122
Captura de pantalla de proceso de creación de mamposterías 4. 122

Figura 59 123
Datos de salida obtenidos en el algoritmo de Reconstrucción. 123

Figura 60 124
Plano obtenido en base al boceto inicial. 124

Figura 61 128
Espesores de losas. 128

Figura 62 128
Condiciones de apoyo de losas. 128

Figura 63 128
Momentos y Cortantes de losas. 128

Figura 64 129
Gráfico de Losa 0. 129

Figura 65 129
Gráfico de losa 1. 129

Capítulo 1

INTRODUCCIÓN

La inteligencia artificial (IA), es un conjunto de algoritmos que tratan de imitar o replicar el funcionamiento del cerebro humano.

Partiendo de que existen diversos tipos de inteligencia, como la memoria y la capacidad de cálculo, en la informática, estas actualmente no son muy sorprendentes, ya que sabemos de la potente capacidad de almacenamiento y rapidez en la resolución de problemas matemáticos, que poseen las computadoras. Lo destacable está en la capacidad de aprendizaje. Dónde por medio de procesos, dando entrada a datos conocidos, se entrena a la IA para la función que se desee que posea. Con este entrenamiento, el sistema obtiene la capacidad de resolver situaciones conocidas y nuevas.

La IA, es aplicable a varias situaciones, tales como:

Reconocimiento de imágenes estáticas, Mejoras del desempeño de la estrategia algorítmica comercial: enfocadas en el sector financiero, Procesamiento eficiente

y escalable de datos de pacientes: para una mejor atención médica, Mantenimiento predictivo: Aplicable en el sector industrial, Detección y clasificación de objetos: Con aplicaciones en distintos sectores, Distribución de contenido en las redes sociales: utilizada como herramienta de marketing, Protección contra amenazas de seguridad cibernética: sirve a los bancos y otros sistemas que operan con transacciones por internet [1].

Al enfocamos en la ingeniería civil, esta al ser una rama clásica, presenta una inmensa oportunidad en el desarrollo de IA en la misma. Este proyecto consiste en desarrollar un método de interpretación de bocetos de planos arquitectónicos, para la detección de los elementos constructivos. Posteriormente por medio de algoritmos, se analizarían y obtendría un archivo CAD del plano en planta, el cómputo de los elementos constructivos, como también el predimensionamiento y la obtención de solicitaciones de las diversas losas que componen a la estructura.

1.1 Planteamiento del problema

Para un ingeniero civil, el tiempo en la realización de planos arquitectónicos, cómputos de los distintos elementos constructivos, como también el cálculo estructural, está basado en la lectura de bosquejos previos hechos en el momento. Este proceso puede ser bastante extenso dependiendo de la envergadura de la construcción a realizar, aún con los softwares disponibles en el mercado actualmente.

El costo de tiempo repercute directamente en el resultado económico, lo cual podría ser mejorado con la implementación de nuevas técnicas que ayuden en las actividades que forman parte de toda la problemática mencionada anteriormente.

Una solución probable es la aplicación de métodos de aprendizaje automático, la cual ha sido poco explorada en ingeniería civil y su mayor implementación en ésta podría ayudar a la reducción de costos económicos y de tiempo. Ésta pasaría a formar parte de las herramientas que ayudarían al ingeniero a aumentar su eficiencia y eficacia en su labor.

1.2 Solución propuesta

El presente proyecto de investigación tiene como objetivo la implementación de métodos de aprendizaje automático mediante la creación de un software basado en esta rama de la inteligencia artificial (IA), que hará lectura de un boceto hecho a mano de un plano arquitectónico, mediante reconocimiento de objetos, identificando los elementos presentes en dicho plano. A partir de las detecciones, por medio de algoritmos, se obtendría como resultado un archivo CAD (.dwg) que contiene el plano en planta, los cómputos de cada elemento constructivo, como también el predimensionamiento de losas y sus solicitaciones.

Con este software se buscará reducir el tiempo implementado por el ingeniero civil para el cálculo o al menos servir de referencia, a la hora de comparar y comprobar resultados.

1.3 Justificación

La semiautomatización de los trabajos de cálculo realizados por el ingeniero puede resultar beneficiosa económicamente debido a un posible aumento en su productividad, eficacia y eficiencia en comparación con los métodos actualmente

utilizados que si bien han mejorado notoriamente con el avance de la tecnología, estos pueden progresar aún más gracias a la implementación de la inteligencia artificial (IA).

Las tecnologías aplicadas a la ingeniería civil continúan en crecimiento y uno de los rumbos que ha tomado en las últimas décadas para dicho crecimiento, ha sido el de la utilización de técnicas de IA, cuya implementación se ha mostrado bastante satisfactoria.

Alemán (2017) menciona algunos ejemplos de estas técnicas:

> Sistemas expertos, utilizados en el diseño de suelos sujetos a cargas axiales, para determinar la capacidad de carga de columnas; redes neuronales artificiales, para pronosticar la fuerza del concreto y sus propiedades con un margen de error aceptable. Sistemas borrosos, utilizados cuando la complejidad del proceso en cuestión es muy alta y los modelos matemáticos existentes no son precisos; y algoritmos genéticos para la predicción de lluvias en un asentamiento. Todas estas investigaciones han demostrado las ventajas de la IA sobre los demás métodos de solución. [2]

Las aplicaciones de las técnicas de IA son relativamente nuevas por lo que aún hay bastantes investigaciones que realizar, en búsqueda de nuevos métodos o actividades dentro de estas técnicas que complementen el trabajo del ingeniero civil y una de ellas es la automatización del reconocimiento de elementos constructivos y estructurales dentro de planos arquitectónicos para su posterior cálculo.

1.4 Objetivos

1.4.1 Objetivo General

Elaborar un método de interpretación automatizada de bocetos arquitectónicos para la obtención de planos en planta CAD, cómputos de los diferentes elementos constructivos, predimensionamiento y solicitaciones de losas.

1.4.2 Objetivos específicos

- Entrenar al modelo con una base de datos para el reconocimiento de diferentes elementos componentes dentro de un boceto arquitectónico.

- Desarrollar un método de ajuste de las predicciones obtenidas para su posterior análisis.

- Desarrollar un algoritmo para la obtención de planos en planta CAD, a partir de las detecciones.

- Desarrollar un algoritmo para el predimensionamiento de losas y la obtención de sus solicitaciones, a partir de las detecciones.

- Desarrollar un algoritmo para la obtención de cómputos de los distintos elementos constructivos, a partir de las detecciones.

- Realizar una comparación entre el método convencional y el método desarrollado teniendo en cuenta los factores de costos y tiempo.

1.5 Antecedentes

- **Fatemi, Nikolic & Moreno (2019)**, en su artículo "Modelling a new workflow based on emotional analysis of floor-plans using machine learning algorithams and semiotics" desarrollan un algoritmo de inteligencia artificial el cual aprende la relación entre el diseño de un plano arquitectónico en planta y las emociones del usuario final dentro de dicho diseño, con el objetivo de crear el más eficiente. Este proyecto tiene la particularidad de realizar una interpretación de planos arquitectónicos, identificando sus diferentes elementos (ventanas, puertas, etc.), base de la cual parte el presente trabajo [3].
- **Massiris, Delrieux & Fernández (2018)**, en su artículo "Detección de equipos de protección personal mediante red neuronal convolucional YOLO" se centran en la monitorización de la presencia y el uso de equipos de protección personal por parte de los trabajadores de la construcción, mediante la utilización de la red neuronal convolucional denominada "You Only Look Once" (YOLO) y que demuestra ser bastante eficiente en cuanto a precisión y tiempo de procesado [4].
- **Muñoz Manso (2014),** en su tesis de grado "Sistema de visión artificial para la detección y lectura de matrículas" desarrolla un algoritmo para la inspección dinámica y automática de placas de matrículas, extrayendo los caracteres de la misma, para su posterior utilización. Esta aplicación tiene un porcentaje de acierto elevado, utilizando una extensa base de datos de caracteres proporcionada por Tesseract [5].
- **Berger, Tamada, & Feltan (2019)**, presentan su artículo "Aplicación móvil para la identificación de variedades de Manihot esculenta Crantz cultivadas en Misiones

mediante técnicas de deep learning" en donde plantean el uso de una aplicación móvil utilizando técnicas de aprendizaje profundo para la identificación de la variedad correspondiente de plantas de mandioca mediante el análisis de una imagen de la hoja o del tallo. Dentro de este proyecto se menciona la utilización del modelo de aprendizaje por transferencia, técnica que será utilizada en el presente trabajo y que reduce considerablemente el entrenamiento de la base de datos que será proporcionada [6].

- **Zambrano Salazar (2017)**, En su proyecto de investigación titulado "Elaboración de un software en base a Matlab para el cálculo y diseño de losas macizas y nervadas" se realizó la escritura de código de un software para el dimensionamiento de losas mediante el uso del programa MATLAB, utilizando de base la Norma Ecuatoriana de la Construcción y el American Concrete Institute [7].

1.6 Metodología

Primeramente se optó, para este trabajo, por un sistema de fácil implementación llamado YOLO (You Only Look Once),

Sistema de código abierto del estado del arte para detección de objetos en tiempo real, el cual ofrece un rendimiento mucho mayor en varios parámetros con respecto a los métodos investigados, ya que la red neuronal puede lograr una velocidad de ejecución de 45 fotogramas por segundo (fps) en computadoras de propósito general. [2]

En vista a que no se disponía de una base de datos conocida, se creó una base de datos propia que constará de una gran cantidad de imágenes de los elementos que componen los planos arquitectónicos (puertas, ventanas, losas, etc.) especificados y etiquetados para su posterior reconocimiento. Esta base de datos fue utilizada para el entrenamiento. Una vez entrenado, se midió la eficacia y eficiencia del detector de imágenes con planos de prueba y se reentrenó, hasta alcanzar una precisión aceptable.

Se desarrollaron algoritmos compatibles, utilizando los resultados obtenidos para el predimensionamiento de losas y obtención de solicitaciones, como también el cómputo de los distintos elementos constructivos y planos en planta CAD.

Todos los algoritmos de este proyecto fueron desarrollados en Python 3.

1.6.1 Creación de Base de datos

Se desarrolló un algoritmo de creación de bocetos que simulan ser a mano con los objetos encuadrados y etiquetados. Tales objetos son elementos que componen usualmente un plano arquitectónico (paredes, puertas, ventanas, etc).

1.6.2 Entrenamiento del detector y prueba

La arquitectura de detección de objetos que se utilizó es YOLO, ya que es de fácil implementación. Con el mismo se realizó el entrenamiento hasta que el detector arrojó buenos resultados en cuanto a la tarea prevista.

1.6.3 Elaboración de algoritmo para corrección de coordenadas

Se desarrolló un algoritmo, que utiliza los elementos detectados y ubica cada uno de ellos con las coordenadas exactas en la posición que les corresponda dentro del espacio de dibujo.

1.6.4 Elaboración de algoritmos para creación de planos, cómputos y cálculo de losas

Se desarrollaron algoritmos, los cuales a partir de las coordenadas obtenidas, procede a la creación de planos CAD (.dwg), creación de cómputos de diversos elementos constructivos (mamposterías, pisos, revoques, etc.) y cálculo de losas, específicamente un predimensionado y obtención de las solicitaciones (momentos y cortantes).

1.6.5 Normas

Para este proyecto se adoptaron las normas ACI, para el uso de tablas entre otras consideraciones para el cálculo de losas y la CIRSOC, específicamente para el uso de pesos específicos y cargas de distintos materiales de construcción.

- Reglamento para concreto estructural. [3]
- Código de construcción para hormigón reforzado. [4]
- Reglamento de cargas permanentes y sobrecargas mínimas. [5]

1.7 Alcance

- El proyecto se limita a construcciones de hasta 200 m2 de superficie. Debido a que a cuanto mayor la superficie, mayor es la distancia a la cámara, disminuyendo la calidad de la imagen, por ende se reduce la precisión de las detecciones.
- Para la foto del boceto, solo se permite utilizar hojas blancas y limpias, con trazados lo más prolijos posibles en bolígrafo azul o negro. Con buena iluminación.
- Las cotas deben estar alineadas a las paredes.

- Las mamposterías inician y terminan con un punto, cada punto representa un pilar y sobre cada mampostería se encuentra una viga. Por ello solo se limita a construcciones cerradas.
- El programa no tiene en cuenta a las mamposterías diagonales.
- El programa no tiene en cuenta a las mamposterías curvas.
- Las puertas son de una sola ala.
- Las mamposterías son todas de un espesor específico que adopta el usuario.
- Los espesores y las alturas de las mamposterías, las secciones transversales de las vigas, pilares y losas se deben cargar manualmente.
- Los pesos de los techos, los volúmenes de mamposterías sobre las losas y sobrecargas se deben de cargar manualmente.
- Las losas deben de están rodeadas por vigas perimetrales (No se calculan voladizos) y se acepta hasta un máximo de 8 metros de luz por losa.
- En el cálculo de losas. Se hallan el predimensionado del espesor y las solicitaciones (Momentos y Cortantes).
- En el cálculo de cómputo. Se hallan las medidas de limpieza y replanteo, mamposterías, aislaciones, revoques interiores y exteriores, pinturas, contrapisos, pisos, zócalos, revestimientos (azulejos), aberturas (puertas y ventanas), hormigón (losas, vigas y pilares).
- Se obtiene el plano en planta CAD (.dwg), este contiene las mamposterías, puertas y ventanas.

Capítulo 2

CONCEPTOS PRELIMINARES

2.1 Aprendizaje Automático

El Machine Learning o Aprendizaje Automático es una rama de la informática en la que se programan sistemas, para que estos puedan aprender por sí solos, aún sin que estas hayan sido programadas específicamente para la situación en particular.

Existen varios tipos de Aprendizaje Automático, Rouhianen (2018) explica la diferencia entre 3 clases [1]:

- Aprendizaje supervisado: Donde los algoritmos utilizan datos que ya han sido etiquetados con anterioridad para indicar la forma que debe ser categorizada la nueva información. Este método, necesita intervención humana.
- Aprendizaje no supervisado: En el cual los algoritmos no utilizan datos etiquetados previamente para indicar cómo debe ser categorizada la nueva información, los mismos algoritmos deben de encontrar la manera de clasificar la información. De tal manera, este método no necesita la intervención humana.

- Aprendizaje por refuerzo: Los algoritmos aprenden de la experiencia. Se le debe de proporcionar un refuerzo positivo las veces que aciertan.

  En este proyecto, se utiliza el tipo de aprendizaje supervisado. Donde se creó una base de datos con elementos etiquetados como puertas, mamposterías, entre otros. Seguidamente se procedió al entrenamiento, para que el sistema reconozca los distintos objetos. Posteriormente se realizó pruebas para ver la capacidad para detectar los elementos correctamente.

2.2 Redes Neuronales

Las redes neuronales son algoritmos que imitan el funcionamiento del cerebro biológico, los perceptrones o neuronas artificiales están agrupadas por capas, las mismas realizan un intercambio de información, esta ingresa y luego es procesada hasta dar una salida.

Haykin (1999) Comenta que las redes neuronales se caracterizan principalmente por [12]:

- Adquirir y almacenar conocimiento de la experiencia.
- Tener capacidad de cambiar dinámicamente junto con el medio.
- Poseer un alto grado de tolerancia a fallos, es decir, pueden sufrir daños significativos y continuar teniendo un buen desempeño.
- Tener un comportamiento paralelo, lo que les permite procesar información procedente de otros fenómenos no-lineales.

2.2.1 Arquitectura del perceptrón

El perceptrón o LTU (Unidad lógica de umbral), recibe una serie de entradas, $x_1, x_2, \dots, x_n$ contenidas en un vector X, cada entrada está relacionada a un peso $w_1, w_2, \dots, w_n$, contenidos en un vector W. Se realiza la multiplicación de cada entrada con su correspondiente peso.

Esta sumatoria que corresponde al cuerpo de la neurona, suma todas las entradas ponderadas, produciendo una salida llamada E. [6]

$$E = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n \quad (1)$$

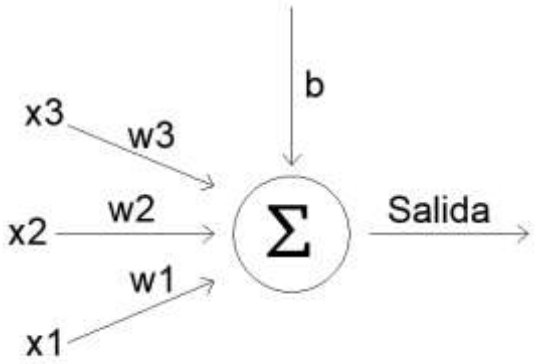
Escrita de forma vectorial

$$E = X^T W \quad (2)$$

Además se le suma un sesgo b (bias) que permite desplazar la función de activación F.

Figura 1

Arquitectura de Perceptrón



2.2.2 Función de Activación

La función de activación F, es la encargada de transmitir la información producida por la suma ponderada y distorsionarla para crear deformaciones no lineales, debido a que la suma de cada una de las rectas de regresión producida por cada neurona, da como resultado otra recta, esto equivale a evitar que la red de neuronas no colapse a una sola neurona.

Entre los tipos de función de activación más utilizados, se encuentran la función sigmoidea y la hiperbólica, estas funciones poseen un rango continuo en los intervalos [0,1], [-1,1] respectivamente, estas funciones están descritas por las siguientes ecuaciones. [6]

$$f_1(x) = \frac{1}{1 + e^{-x}} \quad (3)$$

$$f_2(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \quad (4)$$

Figura 2

Función Sigmoidea

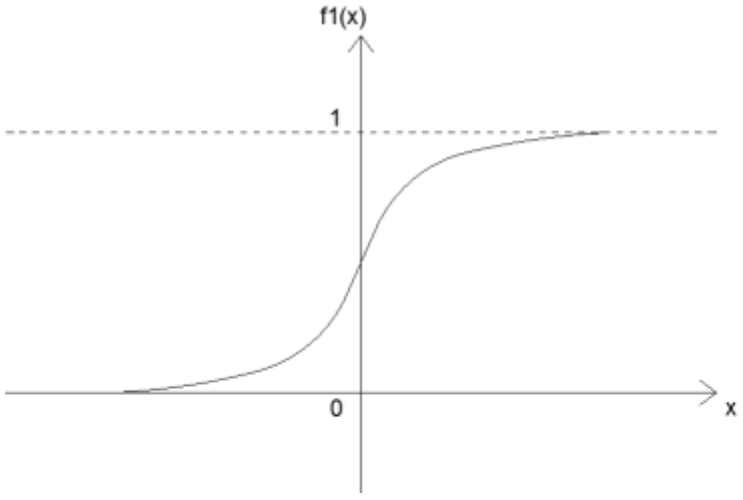
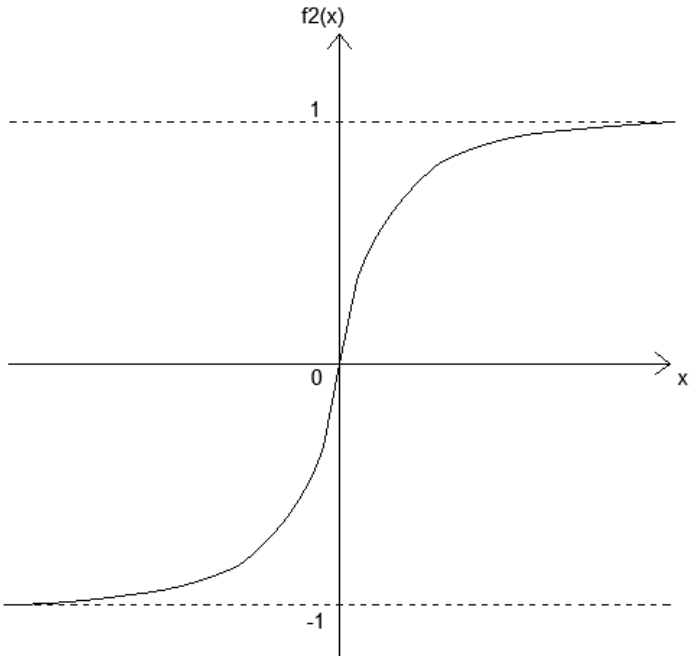


Figura 3

Función Hiperbólica



Otras funciones de activación muy utilizadas son:

La función ReLU (Unidad Lineal Rectificada), que anula los valores negativos y deja los positivos de la misma manera que entraron, transformando así los valores introducidos. Y la función Leaky ReLU (Unidad Lineal Rectificada con fugas), que hace el producto de los valores negativos por un coeficiente rectificador y deja los positivos de la misma manera que entraron, transformando así los valores introducidos [7].

$$f3(x) = \max(0, x) = \begin{cases} 0, & for\ x < 0 \\ x, & for\ x \geq 0 \end{cases} \quad (5)$$

$$f4(x) = \begin{cases} 0, & for\ x < 0 \\ ax, & for\ x \geq 0 \end{cases} \quad (6)$$

Figura 4

Función ReLU

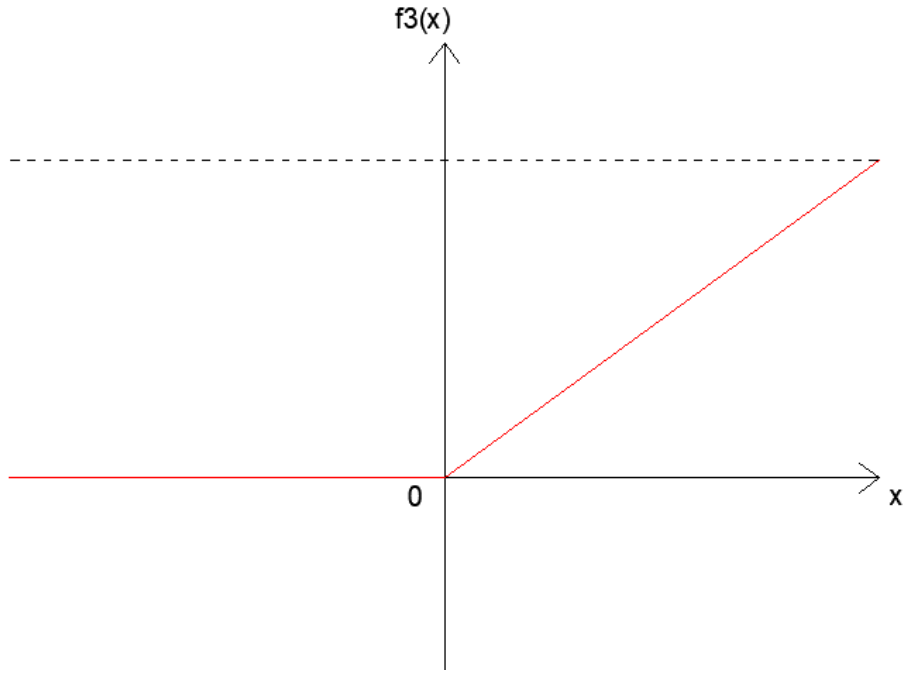
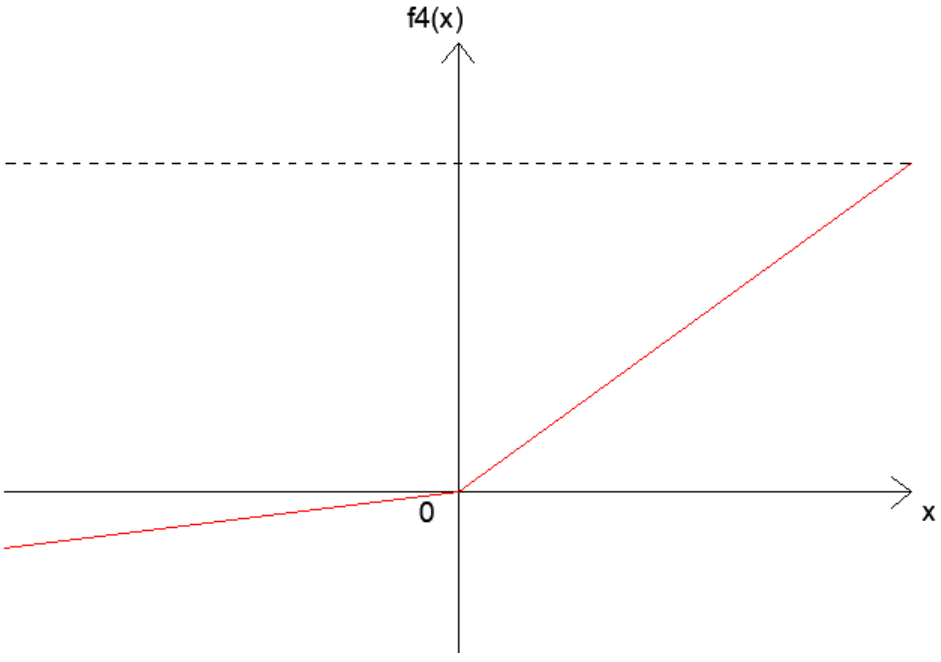


Figura 5

Función Leaky ReLU

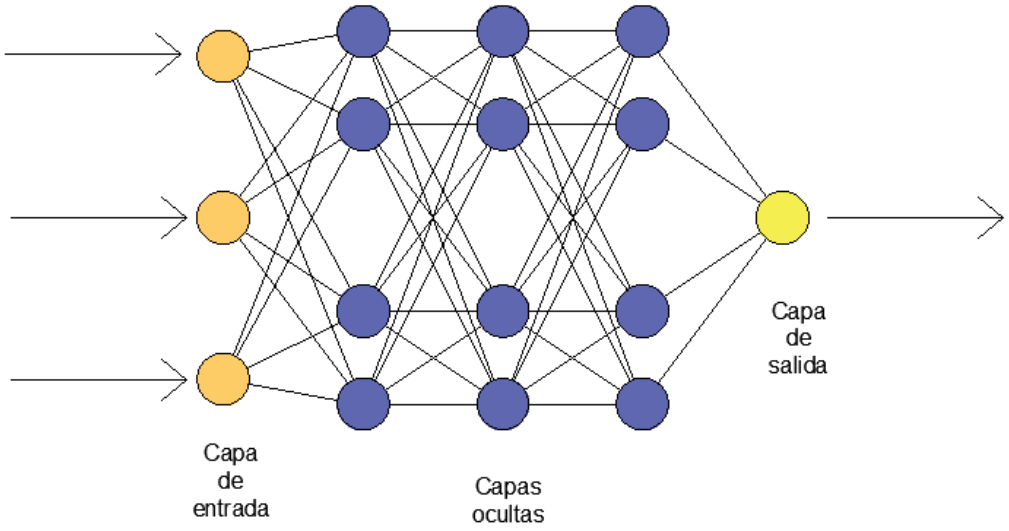


La E es procesada por la función de activación F que produce una señal de salida S, dependiendo de la suma ponderada, si E tiene un valor mayor o igual que el umbral $S = 1$, en caso contrario si E posee un valor menor al umbral $S = 0$, como vimos en el apartado anterior, también existen otros intervalos de salida.

Al obtenerse la salida de una neurona, esta se dispersa ingresando en otras neuronas. La estructura de conexión entre las neuronas es multicapa, entre ellas se distinguen 3 tipos, las capas de entrada, las capas intermedias o capas ocultas y las capas de salida. Representadas en la siguiente figura.

Figura 6

Arquitectura de Perceptrón multicapa



El funcionamiento de la red neuronal se reduce a lo siguiente, partiendo de un vector de entrada, cada neurona toma los valores del vector para procesarlo, seguidamente da una salida que a su vez ingresa en cada una de las siguientes neuronas correspondiente a la siguiente capa, repitiendo el proceso en cuantas capas tenga la red.

Una vez propagada la entrada en toda la red, en la capa de salida, se obtendrán valores de salida ingresados en un vector.

2.2.3 Proceso de aprendizaje del Perceptrón Multicapa

El aprendizaje del perceptrón multicapa consiste en ajustar los parámetros de pesos y umbral con el fin de obtener una salida deseada, minimizando la función de error E.

Dados el vector de entradas $X(n) = (x_1(n), \dots, x_{nc}(n))$, el vector de salidas de la red $Y(n) = (y_1(n), \dots, y_{nc}(n))$, el vector de salidas deseadas $S(n) = (s_1(n), \dots, s_{nc}(n))$ y el vector de umbrales $U(n) = (u_1(n), \dots, u_{nc}(n))$ para un patrón n correspondiente a N número de patrones disponibles.

Isasi Viñuela & Galván León (2004) desriben el proceso de aprendizaje de una red multicapa con los siguientes pasos [13]:

- Se inicializan aleatoriaente con valores cercanos a cero, los pesos y umbrales de la red.
- Se elige un patrón n del conjunto para entrenamiento, $(X(n),\ S(n))$ y se propaga el vector $X(n)$ a la salida de la red con las siguientes escuaciones, obteniéndose un vector salida $Y(n)$.

  Activación de neuronas para capa de entrada

  $$\propto_i^1 = x_i \qquad (7)$$

  Para $i = 1, \dots, n_1$. Donde $\propto_i^1$ es la activación de neuronas para capa de entrada.

  Activación de neuronas para capas de ocultas

$$\propto_i^c = f(\sum_{j=1}^{nc-1} w_{ji}^{c-1} a_j^{c-1} + u_i^c) \quad (8)$$

Para $i = 1, \dots, n_c$ y $c = 1, \dots, C-1$, donde f es la función de activación, C el número de capas, a_j^{c-1} es la activación de neuronas para capa $c-1$.

Activación de neuronas para capa de salida

$$\propto_i^C = y_i = f(\sum_{j=1}^{nC-1} w_{ji}^{C-1} a_j^{C-1} + u_i^C) \quad (9)$$

Para $i = 1, \dots, n_c$. Donde $\propto_i^C$ es la activación de neuronas para capa de salida.

- Se procede a evaluar el error cuadrático cometido por la red con la siguiente ecuación.

$$e(n) = \frac{1}{2}\sum_{i=1}^{nc}(s_i(n) - y_i(n))^2 \quad (10)$$

- Se aplica la regla delta generalizada para modificar los pesos, calculando los valores ∂, para las neuronas correspondientes a la capa de salida, con la siguiente ecuación.

$$\partial_i^C(n) = -(s_i(n) - y_i(n)) f'(\sum_{j=1}^{nC-1} w_{ji}^{C-1} a_j^{C-1} + u_i^C) \quad (11)$$

- Se calculan los ∂ de las neuronas que se encuentran en las capas ocultas, comenzando desde la última capa oculta y propagando hacia atrás los valores hacia la capa de entrada. Para ello se utiliza esta ecuación.

$$\partial_i^{c+1}(n) = f'(\sum_{k=1}^{nc} w_{kj}^c a_k^c + u_j^c) \sum_{i=1}^{nc+1} \partial_i^{c+2}(n) w_{ji}^c \quad (12)$$

- Se modifican los pesos y umbrales utilizando las primeras dos ecuaciones que siguen para los pesos y umbrales de la capa de salida. Luego se utilizan las siguientes dos ecuaciones para los demás parametros de la red.

$$w_{ji}^{C-1}(n) = w_{ji}^{C-1}(n-1) + \propto \partial_i^C(n) \propto_j^{C-1}(n) \quad (13)$$

Para $j = 1, \dots, n_{C-1}$, $i = 1, \dots, n_C$

$$u_i^C(n) = u_i^C(n-1) + \propto \partial_i^C(n) \quad (14)$$

$$w_{kj}^c(n) = w_{kj}^c(n-1) + \propto \partial_j^{c+1}(n) \propto_k^c (n) \quad (15)$$

Para $k = 1, \dots, n_C$, $j = 1, \dots, n_{C+1}$, $c = 1, \dots, C-2$

$$u_j^{c+1}(n) = u_j^{c+1}(n-1) + \propto \partial_j^{c+1}(n) \quad (16)$$

Para $j = 1, \dots, n_{C+1}$, $c = 1, \dots, C-2$

- Se repiten los cinco pasos anteriores para cada patrón de entrenamiento, completando el ciclo de aprendizaje.
- Se evalúa el error con la siguiente ecuación

$$E = \frac{1}{N} \sum_{n=1}^{N} e(n) \quad (17)$$

- Se repiten los siete pasos anteriores hasta que el error sea mínimo.

2.3 Detección de objetos

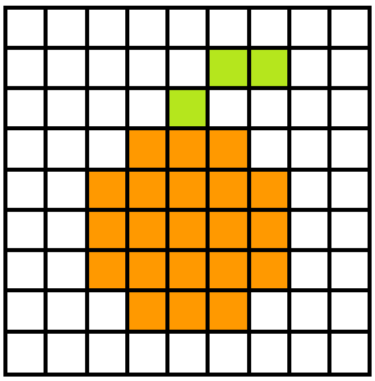
La detección de objetos se basa en una técnica conocida como Visión Artificial, que consiste en un sistema que procesa imágenes para transformarla en caracteres y números con el fin de que la computadora entienda lo que hay en la imagen, es decir, adquiera la capacidad de ver. Con ella podemos detectar la apariencia de objetos dentro de una imagen o una secuencia de imágenes (videos), también podemos obtener el tamaño del objeto, la distancia a la que se encuentra el objeto, etc.

2.3.1 Redes neuronales convolucionales

Partiendo de la imagen, que no es más que una matriz de pixeles (menor unidad homogénea dentro de una imagen). Sus valores están en un intervalo de 0 a 255, El 0 indica el tono más oscuro de un color y 255 la máxima representación de ese color en ese punto. Estos valores al normalizarse para la red neuronal, sus valores que van de 0 a 1.

Figura 7

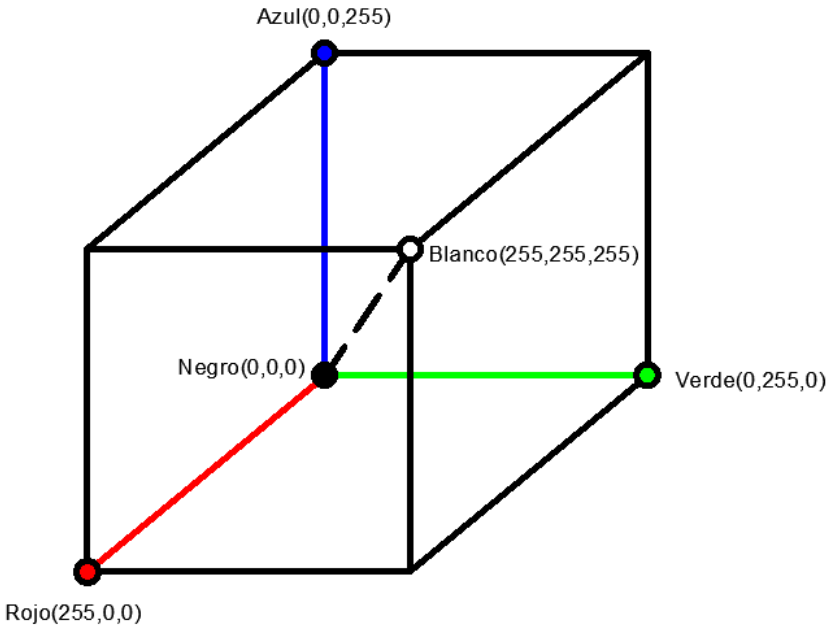
Imagen de una naranja (9 pixeles por 9 pixeles).



En la capa de entrada, para el caso de tener un solo color la red toma como valores de entrada los pixeles de la imagen. Si se tiene más de un color, toma la cantidad de pixeles multiplicada por tres, ya que existen tres canales de color, los cuales son, rojo, verde y azul (RGB).

Figura 8

Representación de Colores RGB en Espacio Tridimensional



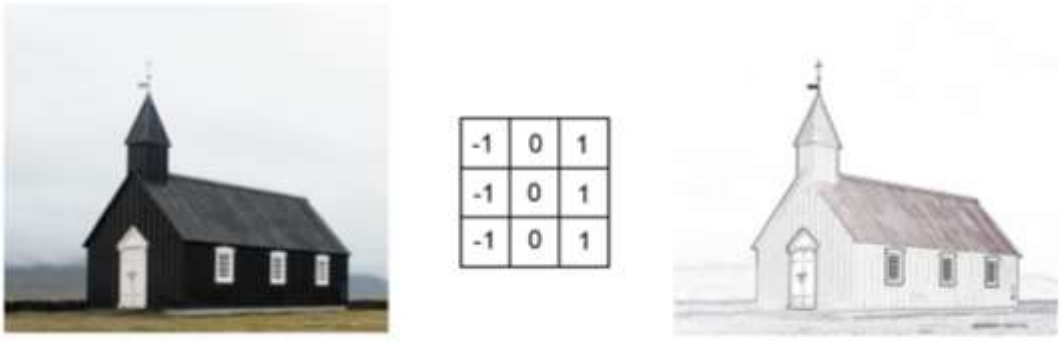
La convolución ocurre cuando, se toman grupos de pixeles cercanos de la imagen y se va operando con un producto escalar con una matriz llamada kernel, esta matriz es de dimensión nxn en caso de ser escala de grises o nxnxn en caso de ser a color.

A la imagen se le aplican filtros, que son conjuntos de kernels. Estos kernels recorren de izquierda a derecha, de arriba hacia abajo cada una de las neuronas de entrada y genera una matriz de salida, que en definitiva será nuestra nueva capa de neuronas ocultas. A estas matrices se las conoce como mapas de características, se las

nombra de esa manera porque sirve como un mapa donde se puede observar la característica buscada por los filtros.

Figura 9

Ejemplo de Filtro para Detectar Bordes.



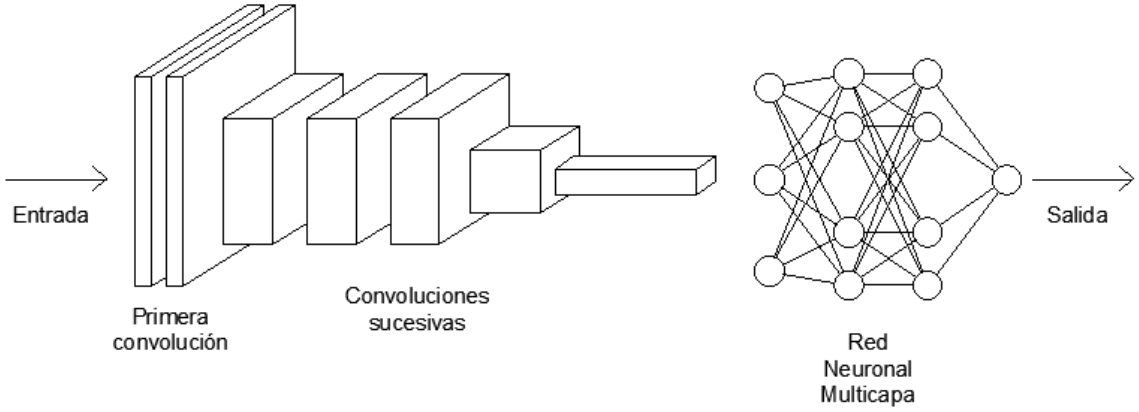
Nota. Foto Skull Kat. Año 2017. Fuente: Unsplash.com.

En las siguientes capas se siguen aplicando filtros, la imagen va disminuyendo su resolución y va aumentando el número de mapa de características. Luego al final se llega a una instancia donde la red convolucional ya detecto los suficientes patrones necesarios y mapas de características, que sirven de entrada a una red neuronal multicapas que se encarga de decidir que objeto está dentro de la imagen.

En síntesis. La función principal de una red neuronal convolucional es aprender los filtros para detectar los patrones que luego nos servirán para detectar uno o varios objetos específicos en una imagen.

Figura 10

Arquitectura de Red Neuronal Convolucional.



2.3.2 Entrenamiento

2.3.2.1 Creación de Base de datos

Primeramente se delimita con un recuadro los objetos dentro de la imagen y se los etiqueta de acuerdo a su clase, esto sirve para indicar al sistema donde se encuentra el objeto, para que pueda aprender. Al haber etiquetado en su totalidad, se obtiene la imagen con los recuadros delimitadores y un fichero que contiene las clases de cada objeto.

Cabe destacar que el proceso de etiquetado es laborioso, ya que se necesita un número elevado de imágenes para entrenar al sistema.

El siguiente paso es dividir el conjunto de datos en tres grupos:

- Entrenamiento, que al cargarlos nos permite adiestrar al sistema para la situación que precisamos.

- Validación, sirve para supervisar al sistema durante su entrenamiento y hacer ajustes de ser necesario.
- Test, son los datos destinados para saber qué tan preciso es el sistema en la detección de objetos.

2.3.2.2 Configuraciones previas

El primer paso es configurar las épocas del entrenamiento, una época el número de veces que se recorren todos los datos, es decir que el entrenamiento tardará las épocas con las que esté configurado el sistema.

Seguidamente tenemos que definir las clases de objetos, es decir los diferentes tipos de objetos a detectar. También se debe de configurar el tamaño de lotes, que representa a una parte de los datos que se va ingresando, para actualizar los pesos mientras ocurre una época.

No se debe dejar de tener en cuenta la estrecha relación que hay entre el tamaño de pasos que el sistema hace para encontrar una solución fiable (Tasa de aprendizaje) y el error que existe entre el resultado esperado con el resultado arrojado por el sistema (Función de coste). Ya que al tener una tasa de aprendizaje alta, no se encuentra el mínimo valor de la función de coste, lo que hace que el resultado no sea fiable. En caso contrario si la tasa de aprendizaje es baja, se llega a un mejor resultado, pero al ser pasos más cortos, el aprendizaje es muy lento. Por ello lo ideal es buscar el equilibrio.

Para realizar el entrenamiento y las detecciones, necesitamos elegir una arquitectura de red neuronal. Específicamente para este trabajo se optó por YOLO, ya que es de fácil implementación, pero existen varias opciones que se pueden elegir.

Las redes se pueden entrenar desde el inicio para cada objeto, pero se puede evitar trabajo si se utiliza el aprendizaje que se realizó con varios objetos a una nueva clase de objeto a ser detectado. A esto se le llama Aprendizaje Transferido.

Se procede a compilar el modelo, esto para controlar la función de pérdida, que permite analizar el rendimiento del sistema para procesar los datos y el optimizador, que nos permite manipular parámetros para disminuir las pérdidas de la función de coste.

También se debe de tener en cuenta puntos de control. Para cada época se genera un punto de control, que tienen la función de almacenar los valores de parámetros que se utilizaron durante el entrenamiento.

Hay que tener en cuenta que se debe configurar un parámetro llamado Detenimiento temprano, que tiene la función de detener automáticamente el entrenamiento, si no se reducen las pérdidas después de ciertas épocas.

Por último, teniendo todas las configuraciones anteriores, podemos iniciar el entrenamiento de la red. Durante el mismo, se debe de controlar el subajuste, que ocurre cuando la función que aproxima el modelo de predicción en realidad no se ajusta a los datos, por ello se debe de aumentar la flexibilidad de la función, aumentado el grado del polinomio. Pero se debe tener en cuenta que si se aumenta demasiado, lo que ocurre es un sobreajuste y la función predice de manera errada.

2.3.2.3 Proceso de Detección

Una vez entrenada a la red, cargamos los pesos ajustados en el entrenamiento, luego procedemos a cargar imágenes a la red, para probar la detección de los objetos en el caso que se encuentren en las imágenes.

El sistema comenzará a dibujar recuadros delimitadores alrededor de las clases de objetos definidos previamente en el entrenamiento. Obteniendo la clase de objeto y sus coordenadas en la imagen.

Finalmente se procede a observar el rendimiento en la detección, por medio de métricas incorporadas en el sistema adoptado.

2.4 Python

El lenguaje de programación utilizado para el desarrollo de los algoritmos para el presente trabajo fue Python 3.

Python es un lenguaje de programación de alto nivel, multiplataforma, de propósito general, que tiene como uno de sus fines, mantener un orden para que el código sea legible.

Es un lenguaje libre y gratuito, intuitivo de usar, es de rápido y fácil aprendizaje, como también de fácil lectura. Otra ventaja que presenta, es la gran cantidad de librerías que ofrece. A continuación se mencionan las librerías utilizadas para este proyecto.

- **numpy:** Librería que permite trabajar con vectores y matrices n dimensionales, como también funciones de alto nivel. Desarrollada por Oliphant (2006) [8].
- **pandas:** Librería que permite la manipulación y análisis de datos de forma estructurada. Desarrollada por McKinney (2009) [9].
- **cv2:** Librería de visión artificial. Desarrollada por Intel Corporation (2000) [10].
- **openpyxl:** Librería que permite leer y escribir archivos Excel. Desarrollada por Gazoni & Clark (2013) [11].

- **pytoautocad:** Librería que permite automatizar la creación de dibujos en AutoCAD. Desarrollada por Haritonov (2012) [12].

Seguidamente una breve explicación de Google Colab:

Google Colab o Collaboratory, es un cuaderno escrito en Jupyter notebook, gratuito que se ejecuta en los servidores de la nube de Google. Este no precisa de una instalación como tampoco configuración de Python. En el mismo se dispone de un acceso a hardware CPU, GPU y TPU de manera gratuita [13].

2.5 YOLO

You Only Look Once (YOLO), creada por Joseph Redmond, es una arquitectura de red neuronal extremadamente rápida y precisa, desarrollada para la detección de objetos. Fue creado en entorno personalizado llamado Darknet. Que es un entorno de investigación muy flexible escrito en lenguajes de bajo nivel.

Redmond, Divvala, Girshick & Farhadi (2016), mencionan.

YOLO aprende representaciones generalizables de objetos. Cuando se entrena con imágenes naturales y se prueba en obras de arte, YOLO supera a los mejores métodos de detección como DPM y R-CNN por un amplio margen. Dado que YOLO es altamente generalizable, es menos probable que se rompa cuando se aplica a nuevos dominios o entradas inesperadas [21].

2.5.1 Funcionamiento de YOLO

En una cuadrícula de SxS, YOLO divide la imagen de entrada, si el centro de un objeto cae en una celda de cuadrícula, esa misma celda es la que se encarga de detectar ese objeto.

Cada una de las celdas, predice recuadros delimitadores y puntajes de confianza para cada caja. Estos puntajes indican la confianza del modelo de la celda que contiene al objeto y también la precisión para predecir al objeto. La confianza se define con la siguiente ecuación:

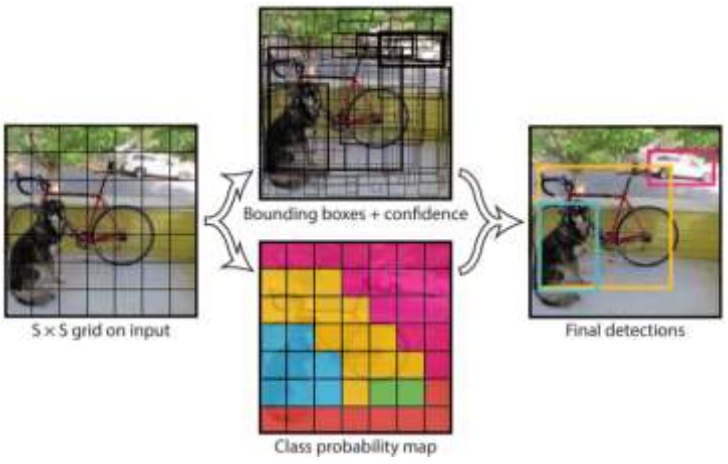
$$Pr(Object) * IOU \quad (18)$$

Si no se encuentra ningún objeto en esa celda, los puntajes de confianza deben ser cero. En caso contrario, si en la celda hay un objeto, necesitamos que el puntaje de confianza sea igual a la intersección sobre unión (IOU).

Cada recuadro delimitador posee 5 parámetros: x, y, w, h, como también confianza. Las coordenadas (x, y) representan el centro del recuadro en relación a los límites de la celda de la cuadrícula. El ancho y el alto se obtienen en relación con la imagen completa.

Figura 11

Detección realizada por YOLO.

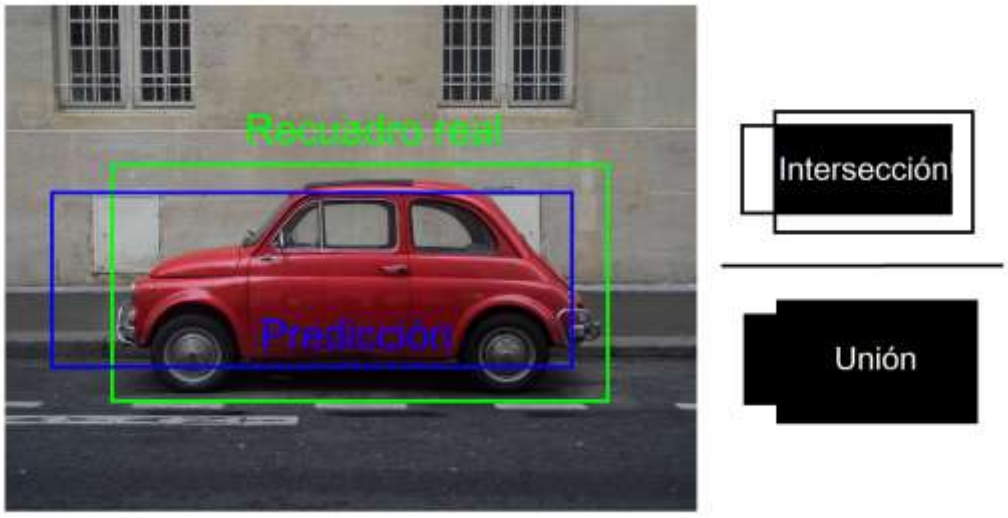


Nota. The model. Año 2016. Fuente: You Only Look Once: Unified, Real-Time Object Detection.

Finalmente la predicción de confianza representa el IOU entre el recuadro predicho y el recuadro real.

Figura 12

Intersección sobre la unión.



Nota. Foto Sean Whelan. Año 2017. Fuente: Unsplash.com.

Las celdas de la cuadrícula predicen las probabilidades de clase condicional de C, descrita por la ecuación:

$$Pr(Class|Objet) \quad (19)$$

La celda de la cuadrícula que contiene al objeto, condicionan dichas probabilidades. Independientemente de la cantidad de recuadros, predice un único objeto.

Cuando ocurre la detección, las probabilidades de clase condicional se multiplican por las predicciones de confianza de un recuadro individual. Como se muestra en la ecuación:

$$Pr(Class|Objet) * Pr(Object) * IOU_{pred}^{truth} = Pr(Class) * IOU_{pred}^{truth} \quad (20)$$

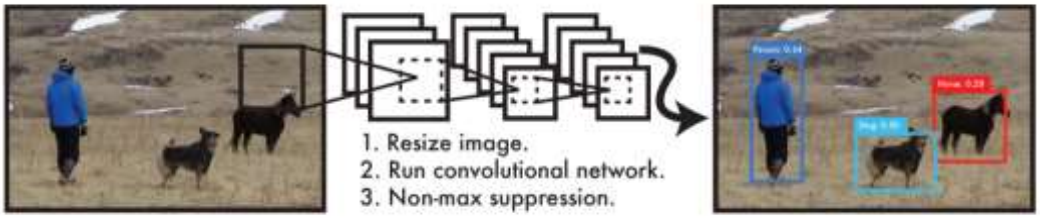
Esta nos proporciona puntajes de confianza específicos de clase para cada recuadro. Estos puntajes muestran tanto la probabilidad de que el objeto se encuentre en el recuadro y qué tan bien se ajusta el recuadro predicho al objeto.

2.5.2 Arquitectura

YOLO está modelado como una red neuronal convolucional. En la primera capa convolucional, se extraen las características de la imagen, seguidamente en las demás capas que se encuentran completamente conectadas predicen las probabilidades de salida como también las coordenadas. La arquitectura de red de YOLO, está inspirada en GoogLeNet que se utiliza para clasificar imágenes

Figura 13

Sistema de Detección de YOLO.



Nota. The YOLO Detection System. Año 2016. Fuente: You Only Look Once: Unified, Real-Time Object Detection.

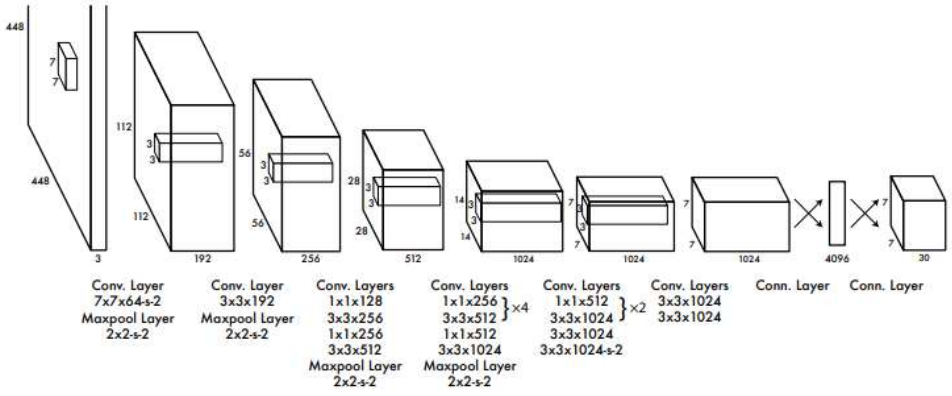
La red tiene 24 capas convolucionales seguidas de 2 capas completamente conectadas. Utiliza capas de reducción de 1×1 seguidas de capas convolucionales de 3×3. Alternando las capas convolucionales 1×1 reducen el espacio de características de las capas anteriores. Al final la red da como resultado es un tensor 7×7×30.

YOLO posee una resolución de entrada de la red 448 × 448. Para aumentar el rendimiento, la red cuenta con cuatro capas convolucionales y dos capas completamente conectadas con los pesos inicializados aleatoriamente.

La capa final predice tanto las probabilidades de clase como las coordenadas del cuadro delimitador. El ancho y el alto del recuadro delimitador, se normaliza de acuerdo al ancho y el alto de la imagen para que tomen valores entre 0 y 1. Del mismo modo se parametrizan las coordenadas x e y del recuadro delimitador para que sean compensaciones de una ubicación de celda de cuadrícula particular, de tal manera que también se encuentra entre los valores de 0 y 1.

Figura 14

Arquitectura de YOLO.



Esta arquitectura utiliza en la capa final una función de activación lineal y en todas las demás capas usan la función de activación Leaky ReLu descrita en la siguiente expresión:

$$\emptyset(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1x, & \text{otherwise} \end{cases} \qquad (21)$$

YOLO utiliza el error de suma cuadrática optimizada para calcular la pérdida, entre los recuadros delimitadores de las detecciones y los recuadros delimitadores reales. Utilizando la siguiente ecuación:

$$\lambda_{cord} \sum_{i=0}^{S^2} \sum_{J=0}^{B} \mathrm{I}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \quad (22)$$

$$+ \lambda_{cord} \sum_{i=0}^{S^2} \sum_{J=0}^{B} \mathrm{I}_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \quad (23)$$

$$+ \sum_{i=0}^{S^2} \sum_{J=0}^{B} \mathrm{I}_{ij}^{obj} \left(C_i - \hat{C}_i \right)^2 \quad (24)$$

$$+ \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{J=0}^{B} \mathrm{I}_{ij}^{noobj} \left(C_i - \hat{C}_i \right)^2 \quad (25)$$

$$+ \sum_{i=0}^{S^2} \mathrm{I}_{i}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \quad (26)$$

2.5.3 Evolución de YOLO

2.5.3.1 YOLO v2

YOLO v2 (YOLO 9000), fue el algoritmo para detección de objetos más rápido y uno de los más precisos. Sin embargo, con los años dejó de ser el más preciso siendo superado por otros detectores, aunque seguía siendo uno de los más rápidos.

YOLO v2 utilizaba darknet-19, una red con 30 capas, 19 capas con 11 capas más para la detección de objetos. Esta versión tuvo varios problemas con la detección

de objetos pequeños. Esto se atribuyó a la pérdida de características de grano fino a medida que las capas reducían la muestra de la entrada. Para remediar esto, se utilizó un mapeo de identidad, concatenando mapas de características de una capa anterior para capturar características de bajo nivel. Pero esta versión aún no poseía algunos de los elementos más importantes que en la actualidad son básicos. [14]

2.5.3.2 YOLO v3

YOLO v3 utiliza una variante de Darknet, que posee una red de 53 capas entrenada en Imagenet. Para la tarea de detección, se apilan 53 capas más, lo que nos brinda una arquitectura subyacente totalmente convolucional de 106 capas para YOLO v3. Debido a esto YOLO v3 es más lenta que YOLO v2. La velocidad fue sacrificada para mejorar la precisión. Dejó de ejecutarse a 45 FPS, pasando a 30 FPS.

Redmon & Farhadi, (2018) comentan [23]:

Esta versión hace predicciones en tres escalas, que se dan precisamente al reducir las dimensiones de la imagen de entrada en 32, 16 y 8 respectivamente.

En la capa 82 ocurre la primera detección. Para las 81 capas anteriores, la imagen es muestreada por la red, de modo que la capa 81 tiene un paso de 32. Si tenemos una imagen de 416x416, el mapa de características resultante tendría un tamaño de 13x13. La detección realiza utilizando un kernel de detección 1x1, obteniéndose un mapa de características de 13x13x255.

Luego, en la capa 79, el mapa de características se somete a algunas capas convolucionales antes de ser muestreado por 2 y pasa a tener 26x26 dimensiones.

Este mapa de características luego se concatena en profundidad con el mapa de características de la capa 61. Luego, los mapas de características combinados se someten a algunas capas convolucionales de 1x1 para fusionar las características de la capa 61.

Seguidamente, en la capa 94, ocurre la segunda detección, que genera un mapa de características de detección de 26x26x255. Luego ocurre un procedimiento similar, donde en la capa 91, el mapa de características se somete a unas capas convolucionales antes de concatenarse en profundidad con un mapa de características de la capa 36. Como ocurrió antes, siguen unas pocas capas convolucionales 1x1 para fusionar la información de la anterior. Finalmente en la capa 106 se finalizan los 3 mapas, produciendo un mapa de características de tamaño 52x52x255.

Debido a las detecciones en diferentes capas. Se ve una mejora en la detección de objetos pequeños.

La capa de 13 x 13 es responsable de detectar objetos grandes, mientras que la capa de 52 x 52 detecta los objetos más pequeños, y la capa de 26 x 26 detecta objetos medianos.

La versión v3 predice 10 veces más cuadros delimitadores que la versión v2.

También hubo cambios en la función de pérdida. YOLO v3 utiliza términos de error de entropía cruzada. Es decir, las predicciones de clase y confianza de objetos en YOLO v3 ahora se predicen a través de la regresión logística.

YOLO v3 realiza una clasificación multietiqueta para los objetos detectados en las imágenes. Anteriormente en YOLO, los autores solían maximizar las puntuaciones de la clase y tomar la clase con la puntuación máxima como la clase del objeto contenido en el recuadro delimitador. Se basaba en la suposición de que si un objeto pertenece a una clase, entonces no puede pertenecer a la otra. Sin embargo, cuando tenemos clases como Persona y Mujer en un conjunto de datos, la suposición anterior falla. Por esta la razón los autores de YOLO se han abstenido de suavizar las clases. En su lugar, cada puntaje de clase se predice mediante regresión logística y se usa un umbral para predecir múltiples etiquetas para un objeto.

2.5.3.3 YOLO v4

Con esta versión se consiguió un detector de objetos eficiente y potente. Hace que todos puedan usar una 1080 Ti o 2080 Ti GPU para entrenar al detector de objetos de manera rápida y precisa. Además se verificó la influencia de los métodos de detección de objetos Bag-of-Freebies y Bag-of-Special de última generación durante el entrenamiento del detector. También se modificaron métodos de última generación para hacerlos más eficientes y adecuados para el entrenamiento de una sola GPU, incluyendo CBN, PAN, SAM, etc.

Esta versión se compone por las siguientes partes [15]:

- Backbone: Es la arquitectura de aprendizaje profundo que vendría ser un extractor de funciones. Básicamente todos los modelos de columna vertebral son modelos de clasificación.

Este modelo opta por CSPDarknet53, este eliminan los cuellos de botella computacionales en DenseNet y mejora el aprendizaje sin modificar el mapa de funciones.

- Neck: Es un subconjunto especial, recopila mapas de características de diferentes etapas de la columna vertebral. Es decir, agrega características.

  El modelo usa en el cuello SPP y PAN

  SPP aumenta el campo receptivo y separa las funciones más importantes de la red troncal. La agrupación de pirámide espacial es que tomamos una imagen de entrada y usamos capas convolucionales para extraer su mapa de características, luego usamos el grupo máximo de tamaño de ventana_1 para generar un conjunto de características, luego usamos nuevamente el grupo máximo de tamaño de ventana_2, repita esto *n* veces y así obtener diferentes mapas de características en altura y ancho, por lo que se forma una pirámide. YOLOv4 divide la característica a lo largo de la dimensión de profundidad, aplica SPP en cada parte y luego lo combina nuevamente para generar un mapa de características de salida.

  PAN: PANet es básicamente una red piramidal de funciones que extrae funciones importantes del clasificador principal.

- Head: También se conoce como el detector de objetos, es la encargada de encontrar la región donde el objeto podría estar presente pero no dice qué objeto está presente en esa región. Este modelo usa como cabeza YOLOv3.

2.6 Losas

Las losas de hormigón son placas planas, en donde el espesor de las mismas es mucho menor a las dimensiones de largo y ancho. Estas soportan cargas perpendiculares al área de la misma, Esto causa que la flexión sea predominante.

Existe gran variedad de tipo de losas, entre ellas se encuentran las macizas, alivianadas, nervadas, etc. Pero nos centraremos en hablar de las losas macizas, ya que es lo que abarca al proyecto.

Las losas macizas son aquellas que poseen un espesor totalmente de hormigón, con armaduras colocadas mayormente en la parte inferior, para resistir las solicitaciones por flexión. Este tipo de losa posee la ventaja de ser fácil de ejecutar.

2.6.1 Tipos de losas según su dirección de armado

- Losas armadas en una dirección, son aquellas que la relación entre la dirección mayor y la dirección menor es mayor a dos. Distribuyen las cargas a los apoyos en la dirección paralela al lado menor.
- Losas armadas en dos direcciones, son aquellas que la relación entre la dirección mayor y la dirección menor es menor o igual a dos. Distribuyen las cargas a los apoyos en las dos direcciones.

Figura 15

Losa Armada en una Dirección.

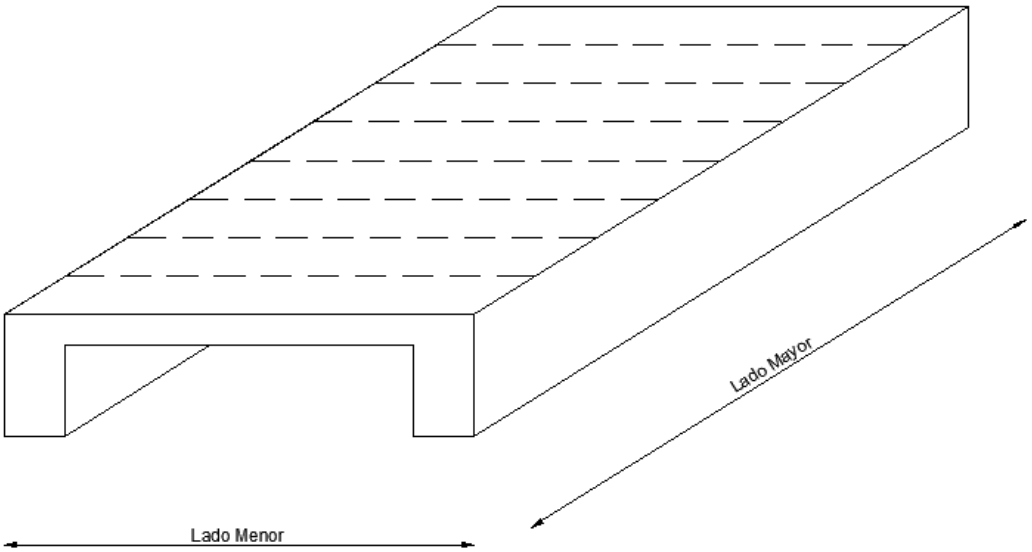
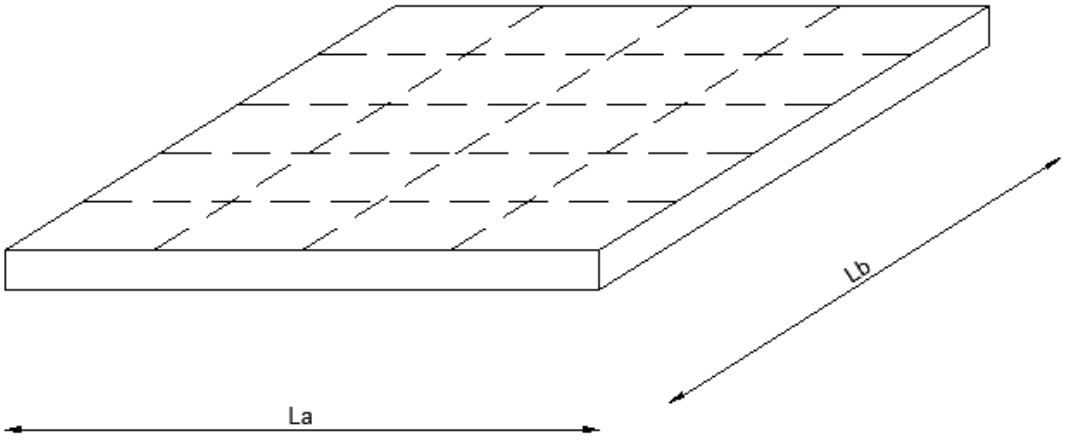


Figura 16

Losa Armada en dos Direcciones.



2.6.2 Tipos de losas según sus apoyos

Para losas con bordes perimetrales:

- Apoyadas en sus cuatro lados.

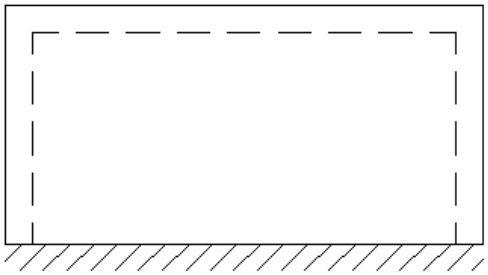
Figura 17

Losa Apoyada en sus cuatro lados.

- Lado menor empotrado.

Figura 18

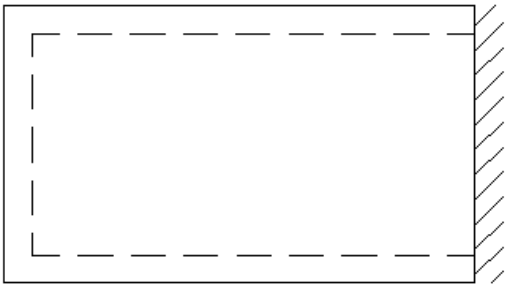
Losa con lado menor empotrado.



- Lado mayor empotrado.

Figura 19

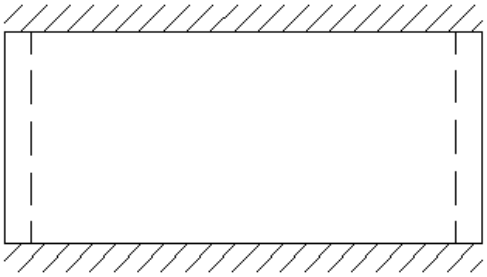
Losa con lado mayor empotrado.



- Los dos lados menores empotrados.

Figura 20

Losa con lados menores empotrados.



- Los dos lados mayores empotrados.

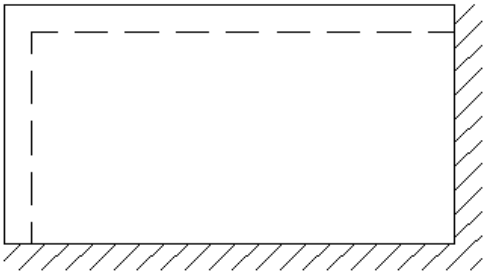
Figura 21

Losa con lados mayores empotrados.

- Los dos lados perpendiculares empotrados.

Figura 22

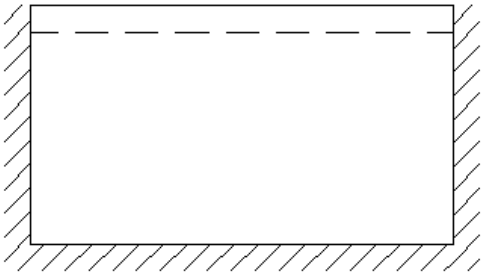
Losa con lados perpendiculares empotrados.



- Los dos lados mayores y uno menor empotrado.

Figura 23

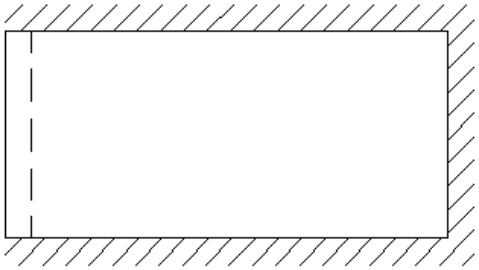
Losa con lados mayores y un lado menor empotrado.



- Los dos lados menores y uno mayor empotrado.

Figura 24

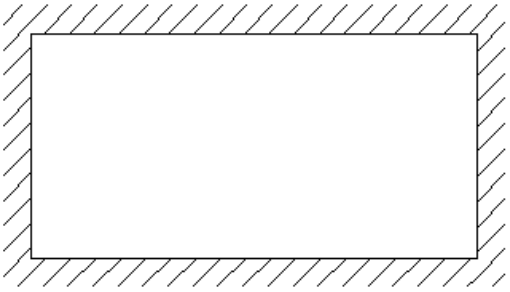
Losa con lados menores y un lado mayor empotrado.



- Los cuatro bordes empotrados.

Figura 25

Losa con cuatro lados empotrados.



2.6.3 Condiciones de apoyo

Como vimos en el apartado anterior, las losas pueden tener distintas combinaciones de apoyo. Hay varias consideraciones que se pueden realizar optar por un tipo de apoyo para cada lado correspondiente a la losa a analizar. Para este proyecto se realizaron las siguientes:

Primeramente se asume que las losas que no poseen continuidad hacia cualquiera de sus lados, se las considera simplemente apoyada sobre las vigas perimetrales, ya que generalmente no se hormigonan monolíticamente con las losas.

En el caso de que la losa tenga continuidad hacia cualquiera de sus cuatro lados, es decir se encuentra al lado de otra losa. Se procede a analizar si las losas están a un mismo nivel, si la losa analizada está a un nivel inferior o superior de la losa que se encuentra a uno de sus lados, se la considera apoyada en ese mismo lado. De lo contrario, se analiza la siguiente consideración.

Si la altura de la losa analizada es mayor a la losa que se encuentra a uno de sus lados, se la considera apoyada en ese mismo lado. En caso contrario, es decir si es menor o igual se analiza lo que sigue.

Si dos tercios de la dimensión en x de la losa analizada es menor o igual a de la dimensión en x de la losa que se encuentra a uno de sus lados, como también si se cumple que dos tercios de la dimensión en y de la losa analizada es menor o igual a de la dimensión en y de la losa que se encuentra a uno de sus lados. La losa analizada se considera empotrada en ese mismo lado. Si no se cumple lo anterior, la losa se considera apoyada.

2.6.4 Tipos de cargas

2.6.4.1 Carga Muerta

Son aquellas las cargas permanentes, es decir, son constantes en su ubicación y valor durante la vida útil de la estructura (peso propio, elementos constructivos, etc).

Los valores de esta carga se especifican en la Norma Argentina [5]. Dichos valores se pueden encontrar en el Apéndice A1.

2.6.4.2 Carga viva

Son aquellas cargas que varían de acuerdo al uso que se le asigna a la estructura, estas cargas no tienen una ubicación exacta sobre la estructura (personas, muebles, etc.)

Los valores de esta carga se especifican en la Norma Argentina [5]. Dichos valores se pueden encontrar en el Apéndice A2.

2.6.4.3 Combinaciones de carga

La sumatoria de la carga viva y la carga muerta debe de proporcionarnos la carga total, esta carga debe de estar mayorada, por ello previo a la suma de las cargas se las multiplica por coeficientes de mayoración. Cabe mencionar que a la carga viva se la multiplica por un coeficiente mayor que a la carga muerta porque la carga viva no puede ser prevista exactamente.

La combinación de carga adoptada es la que se encuentra en el capítulo 9.3 del código ACI 318 (2005) [3].

$$U = 1{,}2D + 1{,}6L \qquad (27)$$

Donde U es la carga última mayorada, D es la carga muerta y L la carga viva.

2.6.5 Especificaciones para predimensionamiento y cálculo de solicitaciones

En el capítulo 13.5 del código ACI 318 (2005) nos dice [9]:

Se permite diseñar un sistema de losas mediante cualquier procedimiento que satisfaga las condiciones de equilibrio y compatibilidad geométrica, si se demuestra que la resistencia de diseño en cada sección es por lo menos igual a la resistencia requerida y se cumplen todas las condiciones de funcionamiento.

Para las losas unidireccionales, el predimensionamiento de los espesores está dado por las tabla 9.5(a) del capítulo 9.5 del código ACI 318 (2005). Seguidamente para el cálculo de solicitaciones, se calculan utilizando los coeficientes ACI del capítulo 8 del código ACI 318 (2005).

Para las losas bidireccionales, el predimensionamiento de los espesores está dado por el libro de Nilson (1999), que menciona [25]:

En el diseño de losas en dos direcciones apoyadas en sus bordes, el espesor mínimo para losas de este tipo se toma usualmente igual a 1/180 veces el perímetro del panel.

Las solicitaciones para las losas bidireccionales, se obtienen a partir de los coeficientes de las tablas de H. Marcus, que se encuentra en el código ACI 318 (1963).

2.6.6 Cálculo de solicitaciones para losas macizas unidireccionales

Para la obtención de las solicitaciones de este tipo de losas, se utilizó el Método ACI, del capítulo 8 del código ACI 318 (2005), donde se considera a la losa como una

viga con base de un metro que posee una carga perpendicular a su longitud uniformemente distribuida.

Para utilizar este método se deben de cumplir las siguientes condiciones:

- Los miembros son prismáticos.
- Los vanos sean aproximadamente iguales, sin que el mayor de los vanos adyacentes exceda en más de 20% al menor.
- Las cargas son uniformemente distribuidas.
- La carga viva debe ser menor o igual a tres veces la carga muerta.
- Debe haber dos o más vanos.

Estas tablas las podemos encontrar en el Apéndice A3.

2.6.7 Calculo de solicitaciones para losas macizas bidireccionales

Para obtener los valores de los mementos y las cortantes, se procedió a usar las tablas de Marcus, que se tabula a partir de la consideración de que la losa se está formada por dos fajas, longitudinal y transversal, ambas con una base unitaria, la carga superficial sobre la losa está repartida entre las fajas. Se debe de cumplir que las losas están apoyadas sobre vigas en los cuatro lados y las vigas deben de ser de una altura no menor a tres veces el espesor de la losa.

Estas tablas las podemos encontrar en el Apéndice A4.

2.7 Cómputo Métrico

Es la estimación de las cantidades de cada uno de los rubros que forman parte de una obra, como también tareas o servicios que tienen que ver con la misma.

Lo que se traduce a calcular, según la unidad de medida, las cantidades de las tareas a realizar y materiales a utilizar.

Macchia (2009). Nos facilita la siguiente lista, de donde tomamos los rubros habituales que se realizan en una obra y unidades de medida [25]:

2.7.1 Tareas preliminares

- Limpieza del terreno (Gl, m2).
- Vallas o cercos de obra (Gl, m2).
- Replanteo y nivelación (Gl).
- Extracción de árboles (Gl).

2.7.2 Movimiento de suelo

- Desmonte y retiro de suelo vegetal (m2).
- Excavación a máquina con retiro (m3).
- Excavación a mano con retiro, cimientos, etc. (m3).
- Zanjas para cimientos a mano (m3).
- Relleno de suelo en cimientos o bases (m3).
- Compactación de suelo, equipo manual, en capas (m3).

2.7.3 Estructura

- Bases de homigón armado (m3).
- Columnas de H°A° (m3).
- Vigas de H°A° (m3).
- Losas de H°A° (m3).
- Vigas de arriostramiento superior o inferior (encadenados) (m3).

2.7.4 Mamposterías

- Ladrillos comunes en cimientos 0,45/0,30 m (m2),
- Ladrillos comunes en elevación 0,30/0,15 m (m2).
- Dinteles de ladrillos vistos en Sardinel (ml).
- Alfeizares de ladrillos vistos en Sardinel (ml).
- Bloques cerámicos (m2).

2.7.5 Revoques

- Jaharro y enlucido a la cal (m2).
- Jaharro y enlucido exterior (m2).
- Completo exterior (azotado, jaharro y enlucido) (m2).
- Toma de juntas ladrillos vistos (m2).
- Jaharro bajo revestimientos (con impermeable) (m2).

2.7.6 Techos

- De tejas francesa, colonial, portuguesa, holandesa, etc. (m2).
- De chapas de hierro prepintadas (m2).
- De policarbonato (m2).
- Alisado de cemento bajo o sobre membrana, para protección (m2).
- Cielorraso en general (m2).

2.7.7 Contrapisos

- Cascotes de ladrillos sobre terreno 0,12 m (m2).
- Cascotes sobre losas 0,08 m (m2).
- Cascotes con pendiente en azotea (m2).

2.7.8 Pisos

- Cerámicas (m2).
- Mosaicos graníticos (m2).
- Madera especia (m2).
- Alisado de cemento con endurecedor y color (m2).
- Baldosones para veredas o patios (m2).
- Zocalos (ml).

2.7.9 Revestimientos

- Azulejos (m2).
- Cerámica (m2).
- Impermeable en interior de tanques (m2).

2.7.10 Vidrios y Cristales

- Blindex (m2).
- Traslúcidos o fantasías (m2).
- Armado (m2).
- Espejos Float o Vítrea (m2).

2.7.11 Pintura y Empapelado

- Sintético sobre madera o metal (m2).
- Látex en paredes o cielorrasos (m2).
- Látex exterior (m2).
- Barnices especiales (m2).

Capítulo 3

FUNCIONAMIENTO DEL MÉTODO DE INTERPRETACIÓN DE BOCETOS

En este capítulo se explica el funcionamiento del Método de interpretación automatizada de bocetos arquitectónicos, partiendo de las configuraciones especificas realizadas a YOLO para la detección de los objetos de nuestro interés, es decir los elementos componentes en un plano arquitectónicos como lo son las mamposterías, aberturas, cotas, entre otros.

Se explica la creación de la Base de Datos, el Proceso de Entrenamiento, El Algoritmo Ing. Bob, encargado del proceso de la detección de objetos. Seguidamente se explica el Algoritmo de Reconstrucción, que es el encargado de la extracción de los parámetros necesarios de las detecciones obtenidas para posteriormente enviarlos a los demás algoritmos que también son explicados en este capítulo, los cuales son los siguientes:

- Algoritmo para obtención de planos CAD, que se encarga de representar el boceto arquitectónico del plano en planta, pero como un dibujo CAD.
- Algoritmo para obtención de Cómputos, que se encarga de calcular las cantidades de los diferentes elementos constructivos.
- Algoritmo de Predimensionado y Cálculo de Solicitaciones en Losas, que calcula el espesor predimensionado y los Momentos y Cortantes en las losas.

3.1 Configuraciones previas

Se utilizó el cuaderno creado por Nguyen (2020) [16]. El mismo posee una carpeta llamada darknet_for_colab, que es una carpeta de darknet modificada específicamente para adaptarse al entorno de Colab. A diferencia de lo que recomienda en su página, se procedió a descargar directamente a la carpeta darknet_for_colab a nuestro Drive, con el fin de mantener la modificación mencionada a continuación.

Una de las tareas más importantes de este proyecto es el almacenamiento de los datos de los recuadros delimitadores de las detecciones, tarea que el detector YOLO, existente dentro de la carpeta darknet_for_colab utilizada por Nguyen, no realiza. Por este motivo, se debe proceder a la modificación de uno de sus archivos. El archivo que debemos modificar se llama image.c, el cual se encuentra alojado dentro de la carpeta *src* y esta a su vez se ubica dentro de darknet_for_colab.

Agregamos en la línea número 353 del archivo image.c las siguientes líneas de código.

Figura 26

Modificación del archivo image.c.

image.c

```
348  for (i = 0; i < selected_detections_num; ++i) {
349      const int best_class = selected_detections[i].best_class;
350      printf("%s: %.0f%%", names[best_class],    selected_detections[i].det.prob[best_class] * 100);
351
352
353      FILE* fichero;
354      fichero = fopen("/content/drive/MyDrive/YOLOv4_weight/darknet_for_colab_croquis/predictions.txt", "a+");
355        fprintf(fichero,"%d %f %f %f %f\n",
356                selected_detections[i].best_class,
357                selected_detections[i].det.bbox.x,
358                selected_detections[i].det.bbox.y,
359                selected_detections[i].det.bbox.w,
360                selected_detections[i].det.bbox.h);
361        fclose(fichero);
```

- En la línea 354 se crea un archivo de texto llamado prediction.txt.
- La línea 356 escribe el porcentaje de confiabilidad de los objetos detectados del croquis en el archivo de texto prediction.
- Las líneas 357 y 358 escriben las coordenadas del centro del recuadro delimitador de los objetos detectados del croquis en el archivo de texto prediction.
- Las líneas 359 y 360 escriben la base y la altura del recuadro delimitador de los objetos detectados del croquis en el archivo de texto prediction.

3.2 Creación de Base de datos

Debido a la particularidad de los símbolos propuestos, se optó por la preparación de un conjunto de datos propio.

La preparación del conjunto de datos es una tarea extensa cuando se trata del entrenamiento de una inteligencia artificial, por la cantidad de datos que se le debe proporcionar para que pueda devolver resultados precisos en situaciones de la vida real luego de su entrenamiento.

Debido a la gran cantidad de datos que se menciona en el párrafo anterior, se diseñó un algoritmo capaz de realizar bocetos, ordenados de tal manera que se asemejen a bocetos arquitectónicos dibujados a mano. Esto se logra con una cierta cantidad de fotos de líneas y curvas dibujadas a mano que representan a cada elemento constructivo, específicamente; mamposterías, puertas, ventanas y líneas de cotas, como también números y letras realizadas manualmente, específicamente los números del 0 al 9, incluidas la coma o punto indistintamente.

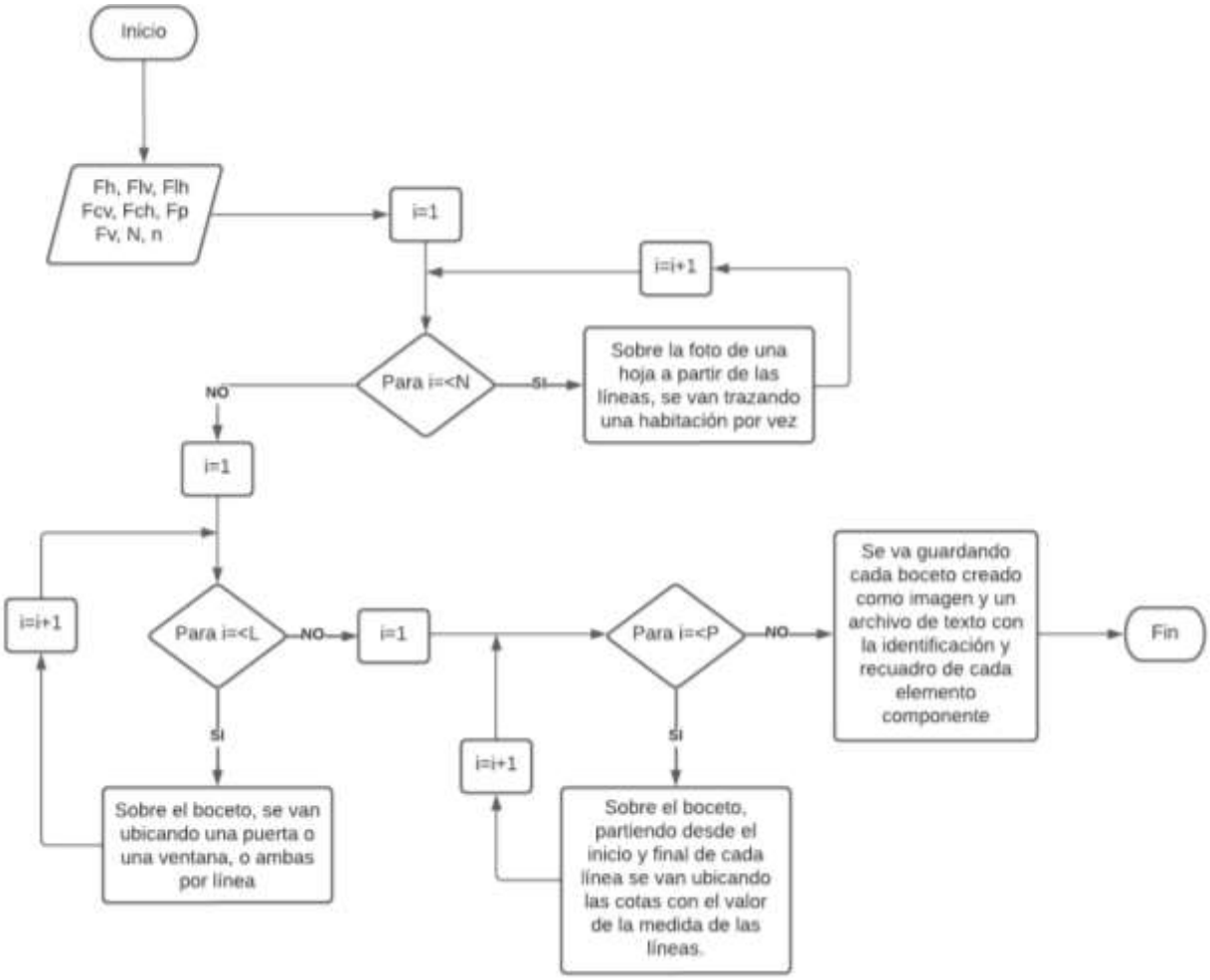
El conjunto de líneas, curvas, letras y números fueron editados por computadora, quitándoles el fondo, dejándolos con un fondo transparente. Para que sean adaptables unos con otros, es decir, se puedan escalar y ordenar automáticamente para formar las paredes de los bocetos y los demás elementos que componen al boceto del plano arquitectónico realizado a mano.

Con este método se acelera el proceso de creación de una base con este tipo de datos. En total se crearon 1000 croquis de este tipo.

El funcionamiento de este algoritmo se puede definir básicamente en el siguiente diagrama de flujo. La explicación más detallada se puede observar en la siguiente página.

Figura 27

Diagrama de flujo de Algoritmo para obtención de bocetos arquitectónicos.



El algoritmo, inicia con fotos de un conjunto que están almacenadas en carpetas, estas imágenes se toman de forma aleatoria y se almacenan en las siguientes variables:

- Fh, son fotos de hojas en blanco.
- Flv, son fotos de líneas verticales con fondo transparente.
- Flh, son fotos de líneas horizontales con fondo transparente.
- Fcv, son fotos de cotas verticales con fondo transparente.
- Fch, son fotos de cotas horizontales con fondo transparente.

- Fp, son fotos de puertas con fondo transparente.
- Fv, son fotos de ventanas con fondo transparente.

También toma valores aleatorios del número de habitaciones a crearse en el boceto definida por la variable N y los números del cero al nueve, como también la coma, definidas en la variable n.

3.2.1 Creación de habitaciones

El algoritmo aleatoriamente toma la foto de una hoja en blanco, contenida en la variable Fh, esta imagen será el lienzo donde procederá a trazar las habitaciones con las fotos aleatorias de las líneas verticales y horizontales. El proceso inicia con un bucle for que va recorriendo la cantidad de habitaciones y por cada habitación, va pegando una línea vertical Flv u horizontal Flh, en cualquier zona de la hoja, seguidamente en sentido horario, pega una línea horizontal o vertical en función inversa a la línea que pegó primero, es decir, si pegó primero una horizontal, luego pegara una vertical y viceversa. A partir de estas dos líneas lo que hará es tomar una copia de las mismas y pegarlas paralelamente para formar el primer rectángulo que será la primera habitación. Luego el algoritmo aleatoriamente partiendo desde cualquier punto de las primeras cuatro líneas trazadas, si tiene el espacio suficiente procederá a pegar otra imagen de línea, de acuerdo a esta línea siguiendo el sentido de las manecillas del reloj, pegará otra línea inversa a la anterior, luego para las siguientes líneas, el algoritmo pegará la foto de otra línea inversa a la anterior y con una sentencia `if/else` analizará si en el espacio de dibujo se encuentra otra línea perpendicular que se cruce en su dirección, aparte a la de su punto de inicio. Si la encuentra, la imagen de la línea se alargará o acortará para adaptarse a la distancia entre el punto de inicio y la línea perpendicular encontrada, así

formando otra habitación. Este último proceso se repetirá hasta alcanzar el número de habitaciones N.

3.2.2 Creación de puertas y ventanas

A partir del boceto de las habitaciones obtenidas, se abre un bucle `for` con sentencias `if/else`. Que recorre el número de líneas del boceto definida por la variable L y va pegando aleatoriamente las fotos de una puerta, una ventana o ambas por línea.

3.2.3 Creación de cotas y sus valores

El algoritmo pega fotos de líneas verticales y horizontales aleatorias que representan a las cotas a cierta distancia de las líneas que representan a las mamposterías, estas toman las medidas máximas de cada lado vertical y horizontal de la construcción, a partir de las mismas se abre un bucle `for` con sentencias `if/else` que van recorriendo las coordenadas de los puntos de inicio y fin de cada línea de mampostería y pegando en las coordenadas trasladadas paralelamente sobre la línea de la cota, una diagonal que delimita las medidas internas de las habitaciones.

Para los números, el algoritmo ubica una coma en el centro de cada medida interna a cierta distancia de las líneas de cotas. A partir de la coma, se ubican a una distancia cercana hacia la izquierda y hacia la derecha los valores de los números aleatorios n.

Una vez completados los procesos anteriores, se obtiene las imágenes de los bocetos arquitectónicos con recuadros delimitadores en cada objeto, con un archivo de texto, que poseen las etiquetas que corresponden a cada objeto, las coordenadas del centro x e y, como también la base y la altura del recuadro.

Por último cabe mencionar que para los identificadores de losas y baños, que son nada más que las letras encerradas en círculos "L" y "b" respectivamente. Se utiliza un algoritmo similar, pero mucho más sencillo. Que no hace nada más que a partir de una foto aleatoria de una hoja, va pegando fotos al azar de los identificadores mencionados, para al final obtener una imagen con su correspondiente archivo de texto, que contienen los mismos parámetros mencionados en el párrafo anterior.

3.3 Entrenamiento

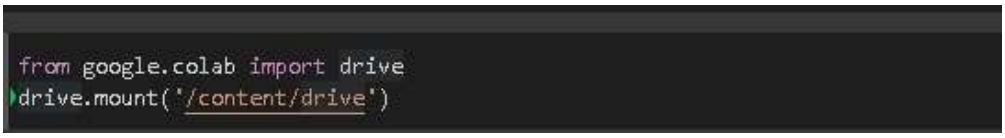
Para entrenar al modelo se utilizó una GPU proporcionada por Google Colab, previo a esto, se realizaron las siguientes configuraciones.

Se habilitó la GPU con anticipación seleccionando Tiempo de ejecución, Cambiar tipo de tiempo de ejecución, Acelerador de hardware, GPU, donde se creó el directorio YOLOv4_weight/backup dentro de Google Drive y se montó Drive en el entorno Colab haciendo lo que sigue.

En una línea de código se escribió lo que sigue y se procedió a ejecutar.

Figura 28

Línea de código que importa Colab a Drive.



Seguidamente se debe de dar la autorización para conectar con drive, donde se debe elegir una cuenta y poner la contraseña. Con este último paso ya es posible utilizar para el entrenamiento la GPU que proporciona Colab.

Figura 29

Ventana de autorización para acceso a Drive.

El modelo se entrenó con YOLOv4, fue necesario un conjunto de datos en formato YOLO. Cada imagen del conjunto de datos está relacionado con un archivo txt con el mismo nombre, que contiene las clases de objetos y sus coordenadas con esta sintaxis (clase de objeto, centro en la coordenada x, centro en la coordenada y, ancho, alto)

Para generar las etiquetas a partir de las imágenes, generalmente se utiliza un programa de etiquetado. Donde básicamente lo que se hace es arrastrar y soltar el mouse para crear un recuadro delimitador alrededor de los objetos. A partir de estos objetos la herramienta genera un archivo de etiquetas. Pero como vimos en el apartado anterior, este paso se realizaba automáticamente con el algoritmo de creación de bocetos arquitectónicos.

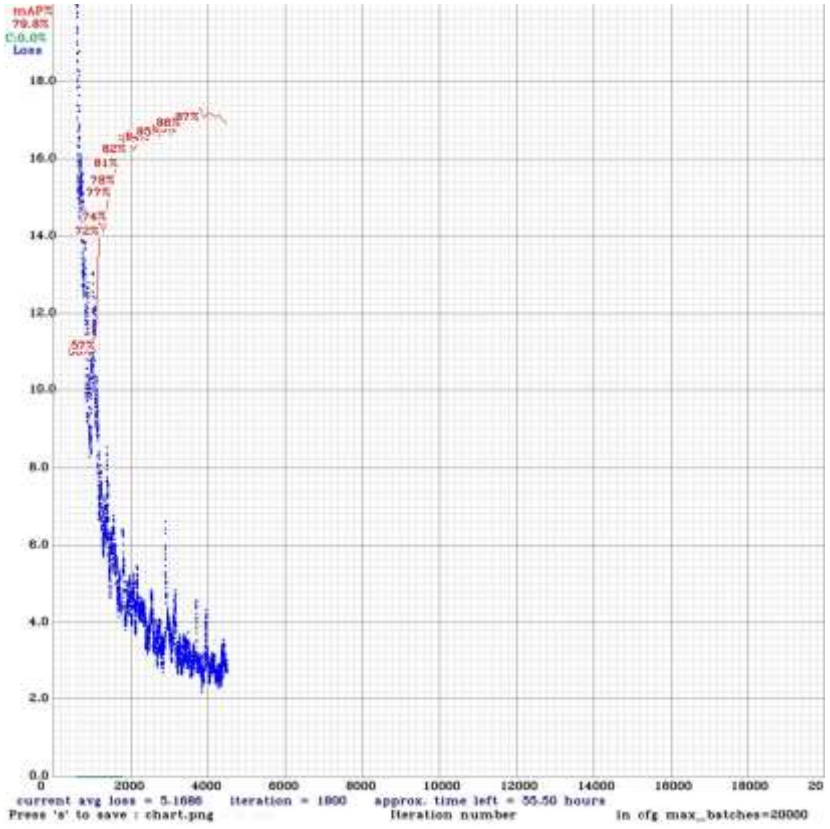
Seguidamente se creó tres archivos de texto que son: classes.names, train.txt y test.txt.

El archivo de classs.names contiene los nombres de los objetos siguiendo este orden: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, Punto, Coma, Pared, Cota, Puerta, Ventana, Sanitario, Losa.

Los archivos de train.txt y test.txt contienen las rutas de acceso de las imágenes de entrenamiento y de prueba respectivamente. Se dividieron los datos en conjuntos de entrenamiento (85 por ciento de los datos) y conjuntos de prueba (15 por ciento de los datos). Esto sirvió para entrenar al modelo en el conjunto de entrenamiento y validarlo en el conjunto de prueba. Se verificó la pérdida del conjunto de entrenamiento, viendo si era alta, lo que significaba que el modelo estaba sufriendo un subajuste, es decir no se ajustaba bien y se necesitaba entrenar más tiempo. En el caso contrario si la pérdida era baja en el conjunto de entrenamiento y alta en el conjunto de prueba, el modelo estaba sobreajustando por lo que se necesitó agregar más datos. Una vez obtenidos resultados aceptables, se dio como concluido el entrenamiento.

Figura 30

Gráfico de porcentajes de aciertos y pérdida del entrenamiento.



3.4 Algoritmo Ing. Bob

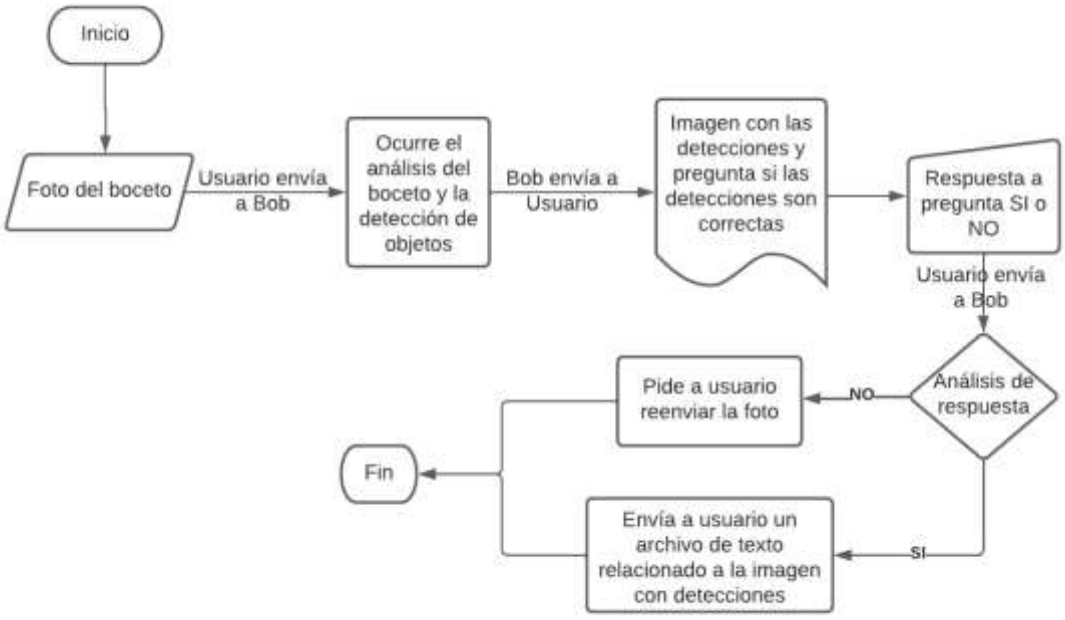
Para una persona poco familiarizada con la informática, el proceso de cargar una imagen a YOLO para que este detecte los objetos, puede resultar algo molesto. Por este motivo se desarrolló un algoritmo al cual nombramos "Ing. Bob". Que tiene la función de ser intermediario entre YOLO y el usuario.

Ing. Bob tiene como base a YOLO, además de una serie de líneas de código que son las encargadas de hacer todo el proceso de carga de imágenes. El algoritmo utiliza Telegram como canal de comunicación entre YOLO y el usuario.

Básicamente el funcionamiento del algoritmo, se describe en el siguiente diagrama de flujo.

Figura 31

Diagrama de flujo de Algoritmo Ing. Bob.



El usuario envía a través de Telegram una foto del boceto arquitectónico o estructural. El algoritmo lo analiza realiza las detecciones de los distintos elementos componentes del boceto y envía la imagen del boceto con las detecciones de los objetos en recuadros delimitadores y sus etiquetas correspondientes a cada objeto, preguntando al usuario si las detecciones son correctas, en caso de que el usuario esté satisfecho con las mismas, responde SI y el algoritmo procederá a enviarle un archivo de texto relacionado a la imagen con las detecciones, dentro del mismo se encuentran las etiquetas con las coordenadas del centro del recuadro x e y, como también la base y la altura del recuadro. En caso contrario, si el usuario responde NO, el algoritmo le pide que vuelva a tomar la foto y se la reenvíe.

3.5 Algoritmo de Reconstrucción

El problema principal a resolver, suponiendo que se realizaron correctamente las detecciones de todos los objetos de interés, es reubicar de manera exacta las mamposterías dentro de un nuevo espacio de dibujo, en el cual las medidas de todos los elementos se encuentran en escala 1:1. Para esto se recurre a la transformación de los datos originales de los cuadros delimitadores, de acuerdo al objeto que contenga cada uno de estos.

3.5.1 Interpretación de los números

Se considera que los dígitos detectados son únicamente los correspondientes a los números que representan las dimensiones de los elementos del plano, que cada número tiene una coma, y que cada número se asocia a una cota en particular.

Con este método de detección, los dígitos y comas son detectados de manera individual, por lo que hay que proceder a agruparlos. Así, por ejemplo, un grupo que forma el número "1,23" (Uno coma veintitrés) dentro de la imagen, tiene como elementos el dígito "1" (uno), el signo "," (coma), el dígito "2" (dos) y el dígito "3" (tres), los cuales se encuentran separados en el momento de la detección. A continuación, se detallan las operaciones matemáticas adoptadas para esta agrupación.

La matriz donde se encuentran las detecciones de los dígitos, tiene la forma:

$$Nd = \begin{bmatrix} C_{Nd}^{(1)} & x_{Nd}^{(1)} & y_{Nd}^{(1)} & w_{Nd}^{(1)} & h_{Nd}^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ C_{Nd}^{(m)} & x_{Nd}^{(m)} & y_{Nd}^{(m)} & w_{Nd}^{(m)} & h_{Nd}^{(m)} \end{bmatrix} \quad (28)$$

Por otro lado, la matriz que contiene los elementos de las detecciones de las comas, tiene la forma:

$$Nc = \begin{bmatrix} C_{Nc}^{(1)} & x_{Nc}^{(1)} & y_{Nc}^{(1)} & w_{Nc}^{(1)} & h_{Nc}^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ C_{Nc}^{(m)} & x_{Nc}^{(m)} & y_{Nc}^{(m)} & w_{Nc}^{(m)} & h_{Nc}^{(m)} \end{bmatrix} \quad (29)$$

La primera tarea consiste en identificar para cada coma dentro de la imagen, los dígitos que se encuentran a una distancia $d_{Nc,Nd}$ menor a un radio r_N, es decir, debe cumplirse la siguiente condición de distancia:

$$d_{Nc,Nd} = \sqrt{\left(x_{Nc}^{(a)} - x_{Nd}^{(b)}\right)^2 + \left(y_{Nc}^{(a)} - y_{Nd}^{(b)}\right)^2} < r_N \quad (30)$$

Para ajustarse al tamaño de los números escritos a mano, el valor de r_N es proporcional a las dimensiones del cuadro delimitador de la coma. Así:

$$r_N = \lambda_N \frac{w_{Nc}^{(i)} + h_{Nc}^{(i)}}{2} \quad (31)$$

Donde λ_N es un coeficiente de ajuste arbitrario que depende del tamaño de las comas con respecto al número al que pertenecen, y de la distancia entre los dígitos que forman los números dentro de la imagen. Se ha utilizado $\lambda_N = 4$ con buenos resultados.

Por cada fila de la matriz Nc construimos una matriz Nn_I de la siguiente manera:

$$Nn_I = \begin{bmatrix} Nc_{(i)} \\ Nd_a \\ Nd_b \\ \vdots \end{bmatrix} = \begin{bmatrix} C_{Nc}^{(i)} & x_{Nc}^{(i)} & y_{Nc}^{(i)} & w_{Nc}^{(i)} & h_{Nc}^{(i)} \\ C_{Nd}^{(a)} & x_{Nd}^{(a)} & y_{Nd}^{(a)} & w_{Nd}^{(a)} & h_{Nd}^{(a)} \\ C_{Nd}^{(b)} & x_{Nd}^{(b)} & y_{Nd}^{(b)} & w_{Nd}^{(b)} & h_{Nd}^{(b)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \quad (32)$$

Donde la primera fila de la matriz Nn_I es igual a una fila de la matriz Nc. Cada fila de la matriz Nd se va añadiendo a la matriz Nn_I que cumpla con la condición de distancia respecto la vector $Nc_{(i)}$, en orden ascendente respecto a la distancia $d_{Nc,Nd}$. Esto último con la finalidad de localizar fácilmente los dígitos que se encuentran a la izquierda y derecha inmediatos de la coma, los cuales se utilizarán posteriormente.

Se supone que los números pueden encontrase en sentido horizontal o vertical dentro de la imagen, y que pueden leerse de izquierda a derecha, y de abajo hacia arriba, respectivamente.

El siguiente paso será ordenar cada matriz Nn_I, de acuerdo a los dígitos representados en cada fila, respecto a su sentido de lectura en la imagen. Para esto, primeramente, debemos determinar si el número correspondiente a matriz se lee en sentido horizontal o en sentido vertical. Se propone realizar esta determinación comparando las distancias entre los límites de los cuadros delimitadores correspondientes a la segunda y tercera fila que representan, como se mencionó anteriormente, los dígitos que se encuentran a la izquierda y derecha inmediatos de la

coma. Se obtuvieron buenos resultados implementando este método empírico para su determinación:

$$Orientación\ (Nn_I) = \begin{cases} Horizontal & \max(D) = d_0 \text{ ó } \max(D) = d_1 \\ Vertical & \max(D) = d_2 \text{ ó } \max(D) = d_3 \end{cases} \quad (33)$$

Donde:

$$D = \{d_0\ , d_1, d_2, d_3\} \quad (34)$$

$$d_0 = \left|\left(x_{Nd}^{(a)} + \frac{w_{Nd}^{(a)}}{2}\right) - \left(x_{Nd}^{(b)} - \frac{w_{Nd}^{(b)}}{2}\right)\right| \quad (35)$$

$$d_1 = \left|\left(x_{Nd}^{(a)} - \frac{w_{Nd}^{(a)}}{2}\right) - \left(x_{Nd}^{(b)} + \frac{w_{Nd}^{(b)}}{2}\right)\right| \quad (36)$$

$$d_2 = \left|\left(y_{Nd}^{(a)} + \frac{h_{Nd}^{(a)}}{2}\right) - \left(y_{Nd}^{(b)} - \frac{h_{Nd}^{(b)}}{2}\right)\right| \quad (37)$$

$$d_3 = \left|\left(y_{Nd}^{(a)} - \frac{h_{Nd}^{(a)}}{2}\right) - \left(y_{Nd}^{(b)} + \frac{h_{Nd}^{(b)}}{2}\right)\right| \quad (38)$$

Una vez determinada la orientación de cada número, se ordena la matriz Nn_I, de acuerdo a dicha orientación: de manera ascendente con respecto a los valores de x de cada fila, en caso de pertenecer a un número horizontal; y de manera descendente con respecto a los valores de y de cada fila, en el caso de tratarse de un número vertical. Cabe mencionar que el ordenamiento de manera decreciente para los números en vertical, se debe a que el sentido de lectura de estos números es contrario al sentido positivo del eje de ordenadas.

Teniendo ordenadas las matrices Nn, se procede a la determinación del valor real del número asociado a cada una. Para facilitar esto se trabaja con la primera columna de la matriz Nn_I:

$$Cn_I = \begin{bmatrix} \cdots & C_{Nd}^{(a)} & C_{Nc}^{(i)} & C_{Nd}^{(b)} & \cdots \end{bmatrix} \quad (39)$$

Se reemplaza cada valor de etiqueta de clase "digito" por el valor del digito correspondiente.

Luego se identifica la posición l de la etiqueta de clase "coma", y se procede a hacer los siguientes cálculos para cada valor $Cn_{I_{(j)}}$:

$$Cn_{I_{(j)}} = \begin{cases} 10^{l-j-1} Cn_{I_{(j)}} & si \quad l > j \\ 0.1^{j-l}\, Cn_{I_{(j)}} & si \quad l < j \end{cases} \quad (40)$$

Finalmente se suman los valores Cn_I y se obtiene el valor del número correspondiente a dicha fila.

$$num^{(i)} = \sum_{j=1}^{n} Cn_{I_{(j)}} \qquad j \neq l \quad (41)$$

Por otro lado, se obtienen las coordenadas de los cuadros delimitadores de cada número como así también sus dimensiones.

Para obtener las coordenadas de los cuadros delimitadores, simplemente se calcula el promedio de las coordenadas de los dígitos:

$$x_N^{(i)} = \frac{x_{Nd}^{(a)} + x_{Nd}^{(b)} + \cdots + x_{Nd}^{(m)}}{m} \quad (42)$$

$$y_N^{(i)} = \frac{y_{Nd}^{(a)} + y_{Nd}^{(b)} + \cdots + y_{Nd}^{(m)}}{m} \quad (43)$$

Mientras que, para las dimensiones de los cuadros delimitadores, se debe tener en cuenta la orientación de lectura del número.

En el caso de los números que se leen horizontalmente, se calcula:

$$w_N^{(i)} = \left| x_{Nd}^{(a)} - x_{Nd}^{(m)} \right| \quad (44)$$

$$h_N^{(i)} = \frac{h_{Nd}^{(a)} + \cdots + h_{Nd}^{(m)}}{m} \quad (45)$$

Mientras que para los números verticales se calcula:

$$w_N^{(i)} = \frac{w_{Nd}^{(a)} + \cdots + w_{Nd}^{(m)}}{m} \quad (46)$$

$$h_N^{(i)} = \left| y_{Nd}^{(a)} - y_{Nd}^{(m)} \right| \quad (47)$$

Donde a y m corresponden a la primera y última fila respectivamente.

Por último, teniendo todos estos valores, se pueden agrupar en una matriz N de la siguiente manera:

$$N = \begin{bmatrix} num^{(1)} & x_N^{(1)} & y_N^{(1)} & w_N^{(1)} & h_N^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ num^{(m)} & x_N^{(m)} & y_N^{(m)} & w_N^{(m)} & h_N^{(m)} \end{bmatrix} \quad (48)$$

Si se realizaron correctamente todas las detecciones, la matriz N debería tener misma cantidad de filas que la matriz donde se encuentran detecciones de las cotas.

3.5.2 Interpretación de las cotas

La matriz donde se encuentran las detecciones de las cotas, tiene la forma:

$$C = \begin{bmatrix} x_C^{(1)} & y_C^{(1)} & w_C^{(1)} & h_C^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_C^{(m)} & y_C^{(m)} & w_C^{(m)} & h_C^{(m)} \end{bmatrix} \quad (49)$$

Primeramente, se asigna a cada cota su valor correspondiente. Para esto, se calculan las distancias entre cada cota y cada número dentro de la imagen, y se le asigna a cada cota el número más cercano a este:

$$Num_C^{(i)} = num^{(m)} \mid d_{i,m} = \min(D_i) \quad (50)$$

Donde:

$$D_i = \{d_{i,0}, d_{i,1}, \dots, d_{i,m}\} \quad (51)$$

$$d_{i,m} = \sqrt{\left(x_C^{(i)} - x_N^{(m)}\right)^2 + \left(y_C^{(i)} - y_N^{(m)}\right)^2} \quad (52)$$

De manera similar a como se vio con los números, las cotas también pueden encontrarse en sentido horizontal o vertical. Por este motivo, la matriz C se divide en dos matrices que contienen las cotas verticales y horizontales por separado, aplicando la siguiente regla:

$$Orientación\ (C_{(i)}) = \begin{cases} Horizontal & si\ \ w_C^{(i)} > h_C^{(i)} \\ Vertical & si\ \ w_C^{(i)} < h_C^{(i)} \end{cases} \quad (53)$$

Una vez determinada la orientación, una de las dimensiones del cuadro delimitador de cada cota deja de tener relevancia, siendo esta, la altura para las cotas horizontales; y el ancho, para las cotas verticales. Entonces, dichas dimensiones pueden eliminarse. Así obtenemos la matriz de cotas horizontales:

$$Ch = \begin{bmatrix} Num_{Ch}^{(1)} & x_{Ch}^{(1)} & y_{Ch}^{(1)} & w_{Ch}^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ Num_{Ch}^{(m)} & x_{Ch}^{(m)} & y_{Ch}^{(m)} & w_{Ch}^{(m)} \end{bmatrix} \quad (54)$$

Y la matriz de cotas verticales:

$$Cv = \begin{bmatrix} Num_{Cv}^{(1)} & x_{Cv}^{(1)} & y_{Cv}^{(1)} & h_{Cv}^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ Num_{Cv}^{(m)} & x_{Cv}^{(m)} & y_{Cv}^{(m)} & h_{Cv}^{(m)} \end{bmatrix} \quad (55)$$

3.5.3 Ubicación de los ejes

La característica principal del método presentado en este trabajo consiste en la ubicación de una línea de construcción en cada extremo de las cotas, perpendicular a estas, las cuales actuarán como ejes de las mamposterías.

Para que el método pueda llevarse a cabo, debe existir una cota entre ejes de mampostería contiguos.

Cada cota tendrá, por lo tanto, dos ejes asociados que pueden entenderse como rectas cuyas ecuaciones son:

$$x = \begin{cases} e_{Ev}^{(i)} = x_{Ch}^{(i)} - \dfrac{w_{Ch}^{(i)}}{2} \\ e'^{(i)}_{Ev} = x_{Ch}^{(i)} + \dfrac{w_{Ch}^{(i)}}{2} \end{cases} \quad \text{En el caso de las cotas horizontales} \quad (56)$$

$$y = \begin{cases} e_{Eh}^{(i)} = y_{Cv}^{(i)} - \dfrac{h_{Cv}^{(i)}}{2} \\ e'^{(i)}_{Eh} = y_{Cv}^{(i)} + \dfrac{h_{Cv}^{(i)}}{2} \end{cases} \quad \text{En el caso de las cotas verticales} \quad (57)$$

Cada eje puede colocarse dentro de un vector $E_{(i)}$:

$$E_{(i)} = \begin{bmatrix} e_E^{(i)} & e'^{(i)}_E & Num_E^{(i)} \end{bmatrix} = \begin{bmatrix} z_C^{(i)} - \dfrac{d_C^{(i)}}{2} & z_C^{(i)} + \dfrac{d_C^{(i)}}{2} & Num_C^{(i)} \end{bmatrix} \quad (58)$$

Donde z puede representar al valor x o al valor y, y d puede representar al valor w o al valor h, dependiendo del tipo de ejes que contenga $E_{(i)}$, como explicaremos a continuación.

Por comodidad a la hora de programar, construimos una matriz de ejes verticales Ev y una matriz de ejes horizontales Eh:

$$Ev = \begin{bmatrix} e_{Ev}^{(1)} & e'^{(1)}_{Ev} & Num_{Ev}^{(1)} \\ \vdots & \vdots & \vdots \\ e_{Ev}^{(m)} & e'^{(m)}_{Ev} & Num_{Ev}^{(m)} \end{bmatrix} = \begin{bmatrix} x_{Ch}^{(1)} - \frac{w_{Ch}^{(1)}}{2} & x_{Ch}^{(1)} + \frac{w_{Ch}^{(1)}}{2} & Num_{Ch}^{(1)} \\ \vdots & \vdots & \vdots \\ x_{Ch}^{(m)} - \frac{w_{Ch}^{(m)}}{2} & x_{Ch}^{(m)} + \frac{w_{Ch}^{(m)}}{2} & Num_{Ch}^{(m)} \end{bmatrix} \quad (59)$$

$$Eh = \begin{bmatrix} e_{Eh}^{(1)} & e'^{(1)}_{Eh} & Num_{Eh}^{(1)} \\ \vdots & \vdots & \vdots \\ e_{Eh}^{(m)} & e'^{(m)}_{Eh} & Num_{Eh}^{(m)} \end{bmatrix} = \begin{bmatrix} y_{Cv}^{(1)} - \frac{h_{Cv}^{(1)}}{2} & y_{Cv}^{(1)} + \frac{h_{Cv}^{(1)}}{2} & Num_{Cv}^{(1)} \\ \vdots & \vdots & \vdots \\ y_{Cv}^{(m)} - \frac{h_{Cv}^{(m)}}{2} & y_{Cv}^{(m)} + \frac{h_{Cv}^{(m)}}{2} & Num_{Cv}^{(m)} \end{bmatrix} \quad (60)$$

Donde cada fila es un vector $E_{(i)}$, que contiene los valores de las cotas horizontales, para el caso de la matriz Ev; y los valores de las cotas verticales, para el caso de la matriz Eh.

Como el procedimiento es el mismo para ambas matrices, podemos hacer los cálculos sobre una matriz E que represente a ambas, indistintamente:

$$E = \begin{bmatrix} e_{E}^{(1)} & e'^{(1)}_{E} & Num_{E}^{(1)} \\ \vdots & \vdots & \vdots \\ e_{E}^{(m)} & e'^{(m)}_{E} & Num_{E}^{(m)} \end{bmatrix} = \begin{bmatrix} z_{C}^{(1)} - \frac{d_{C}^{(1)}}{2} & z_{C}^{(1)} + \frac{d_{C}^{(1)}}{2} & Num_{C}^{(1)} \\ \vdots & \vdots & \vdots \\ z_{C}^{(m)} - \frac{d_{C}^{(m)}}{2} & z_{C}^{(m)} + \frac{d_{C}^{(m)}}{2} & Num_{C}^{(m)} \end{bmatrix} \quad (61)$$

El procedimiento general consiste en convertir la matriz E en una matriz E''', en la que los valores de los ejes coinciden con los ejes de las paredes dentro de la imagen.

Esto lo hacemos reemplazando los ejes que se encuentran cercanos por uno que se encuentre en la coordenada que resulta de promediar las coordenadas de los ejes considerados cercanos.

Para facilitar la programación del método, agruparemos los vectores $E_{(i)}$ de tres maneras distintas.

Primero, agruparemos los vectores cuyo primer elemento, es decir, el valor $e_E^{(a)}$, se encuentren cercanos. Para esto, tomaremos uno de los vectores como referencia. Para considerar que los ejes se encuentran cercanos, se debe cumplir que la distancia entre el eje de referencia y los demás ejes, sea menor a un valor Δ_e:

$$\left|e_E^{(a)} - e_E^{(i)}\right| < \Delta_e \qquad i = 0,1,\dots,m \quad i \neq a \tag{62}$$

Donde Δ_e es un valor arbitrario. Para el presente trabajo se utilizó $\Delta_e = min\,(w_C^{(1)}, h_C^{(1)})$, con buenos resultados.

Entonces, se realiza el promedio entre estos pares de valores y se obtiene el valor para $e_{E\prime}^{(a)}$ y $e_{E\prime}^{(i)}$:

$$e_{E\prime}^{(a)} = e_{E\prime}^{(i)} = \frac{e_E^{(a)} + e_E^{(i)}}{2} \tag{63}$$

Los valores $e'^{(i)}_{E\prime}$ se mantienen iguales a los valores $e'^{(i)}_{E}$.

Esto se realiza con cada uno de los vectores $E_{(i)}$ tomados como referencia.

El resultado es una matriz E' con la forma:

$$E' = \begin{bmatrix} e_{E'}^{(1)} & e'^{(1)}_{E'} & Num_E^{(1)} \\ \vdots & \vdots & \vdots \\ e_{E'}^{(m)} & e'^{(m)}_{E'} & Num_E^{(m)} \end{bmatrix} \quad (64)$$

El segundo agrupamiento se realiza tomando un vector $E'_{(a)}$ como referencia de manera que la distancia entre su primer elemento y el segundo elemento de los demás vectores de E' sea menor al valor Δ_e. Es decir:

$$\left| e_{E'}^{(a)} - e'^{(i)}_{E'} \right| < \Delta_e \qquad i = 0,1,\dots,m \quad i \neq a \quad (65)$$

Entonces, se realiza el promedio entre estos pares de valores y se obtiene el valor para $e_{E''}^{(a)}$ y $e'^{(i)}_{E''}$:

$$e_{E''}^{(a)} = e'^{(i)}_{E''} = \frac{e_{E'}^{(a)} + e'^{(i)}_{E'}}{2} \quad (66)$$

Esto se realiza con cada uno de los vectores $E'_{(i)}$ tomados como referencia.

El resultado es una matriz E'' con la forma:

$$E'' = \begin{bmatrix} e_{E''}^{(1)} & e'^{(1)}_{E''} & Num_E^{(1)} \\ \vdots & \vdots & \vdots \\ e_{E''}^{(m)} & e'^{(m)}_{E''} & Num_E^{(m)} \end{bmatrix} \quad (67)$$

Para el tercer agrupamiento, se toma uno de los vectores $E''_{(i)}$ como referencia y se realiza un procedimiento similar a los anteriores. Esta vez se compara el segundo valor del vector $E''_{(a)}$ con el segundo valor de los demás vectores $E''_{(i)}$.

$$\left| e'^{(a)}_{E''} - e'^{(i)}_{E'} \right| < \Delta_e \qquad i = 0,1,\dots,m \quad i \neq a \quad (68)$$

$$e_{E'''}^{(a)} = e'^{(i)}_{E'''} = \frac{e_{E''}^{(a)} + e'^{(i)}_{E''}}{2} \quad (69)$$

Como resultado obtenemos la matriz E''' con la forma:

$$E''' = \begin{bmatrix} e_{E'''}^{(1)} & e'^{(1)}_{E'''} & Num_{E'''}^{(1)} \\ \vdots & \vdots & \vdots \\ e_{E'''}^{(m)} & e'^{(m)}_{E'''} & Num_{E'''}^{(m)} \end{bmatrix} \quad (70)$$

Cabe destacar que luego de estas transformaciones, cada elemento $Num_E^{(i)}$, mantiene su posición dentro de la matriz E'''. Es decir:

$$Num_{E'''}^{(i)} = Num_E^{(i)} \quad (71)$$

Como se mencionó anteriormente, para este método debe haber una cota entre cada par de ejes contiguos, es decir, para cada par de ejes contiguos debe haber un vector $E'''_{(i)}$ que contenga dicho par de ejes. Puede darse el caso en que la matriz E''' no contenga todos los vectores necesarios para el método, por lo que hay que agregar estos, realizando el siguiente procedimiento.

Primeramente, agregaremos la coordenada real $Cr_{E'''}^{(i)}$ (referida al espacio de dibujo real) del segundo eje de cada vector $E'''_{(i)}$, a cada uno de estos. Para esto ordenamos las filas de la matriz E''' de acuerdo al segundo elemento de cada fila, de manera ascendente, y agregamos cada valor $Cr_{E'''}^{(i)}$ que se define como:

$$Cr_{E'''}^{(i)} = Num_{E'''}^{(i)} + Num_{E'''}^{(i-1)} \quad (72)$$

Cada vector o fila se define ahora como:

$$E'''_{(i)} = \left[e_{E'''}^{(i)} \quad e'^{(i)}_{E'''} \quad Num_{E'''}^{(i)} \quad Cr_{E'''}^{(i)}\right] \quad (73)$$

Se localizan los ejes pertenecientes a mamposterías cuya cara derecha (en el caso de las mamposterías verticales) o cara inferior (en el caso de las mamposterías horizontales), de acuerdo a la disposición de la imagen, se entiende que se encuentra en contacto con el exterior. Podemos identificar estos ejes, observando que únicamente

se encuentran en la segunda posición de al menos un vector $E'''_{(i)}$. Es decir, debe ser un valor $e'^{(a)}_{E'''}$ que no se encuentre en otro vector como valor $e^{(i)}_{E'''}$.

Identificados estos vectores, construimos los vectores relacionados a las cotas faltantes de la matriz $E'''_{(i)}$ de la siguiente manera:

$$E'''_{(m+f)} = \left[e^{(m+f)}_{E'''} \quad e'^{(m+f)}_{E'''} \quad Num^{(m+f)}_{E'''} \quad Cr^{(m+f)}_{E'''}\right] \tag{74}$$

Donde:

$$e^{(m+f)}_{E'''} = e^{(a)}_{E'''} - e^{(b)}_{E'''} \tag{75}$$

$$e'^{(m+f)}_{E'''} = e'^{(a)}_{E'''} \tag{76}$$

$$Num^{(m+f)}_{E'''} = Num^{(a)}_{E'''} - Num^{(b)}_{E'''} \tag{77}$$

$$Cr^{(m+f)}_{E'''} = Cr^{(a)}_{E'''} \tag{78}$$

$$a > b \tag{79}$$

Estos cálculos se realizan tres veces: primeramente, $E'''_{(a)}$ y $E'''_{(b)}$ representan las cotas que contienen los ejes en contacto con el exterior; luego, para la segunda vez que se realizan los cálculos, únicamente $E'''_{(a)}$ representa las cotas que contienen los ejes en contacto con el exterior, mientras que $E'''_{(b)}$ representa las cotas que no contienen dichos ejes; y en la tercera vez, $E'''_{(b)}$ representa las cotas que contienen los ejes en contacto con el exterior y $E'''_{(a)}$, las cotas que no lo tienen.

Se agregarán los nuevos vectores, siempre y cuando no sean iguales a los vectores existentes en la matriz E''', y se ordena nuevamente esta.

Por último, se agrega una cota cuya dimensión es igual a cero, que parte de la última cota de la matriz E'''. Esta cota tiene la forma:

$$E'''_{(m)} = \left[e'^{(m-1)}_{E'''} \quad 0 \quad 0 \quad Cr^{(m-1)}_{E'''}\right] \tag{80}$$

De esta manera, quedan determinadas las cotas dentro de las matrices Ev''' y Eh''' (tratadas anteriormente como matriz E'''). Ambas matrices se utilizan para representar la "grilla" que nos servirá de guía para colocar las mamposterías y aberturas.

Para realizar los siguientes métodos, conviene transformar de antemano cada vector $E'''_{(i)}$ a su homólogo $Er'''_{(i)}$ en el espacio real de dibujo. Definimos esta transformación como sigue:

$$E'''_{(i)} \rightarrow Er'''_{(i)} = \left[e^{(i)}_{Er'''} \quad e'^{(i)}_{Er'''} \quad Cr^{(i)}_{E'''}\right] \tag{81}$$

Donde:

$$e^{(i)}_{Er'''} = Cr^{(i-1)}_{E'''} \tag{82}$$

$$e'^{(i)}_{Er'''} = Cr^{(i-1)}_{E'''} + Num^{(i)}_{E'''} \tag{83}$$

3.5.4 Ubicación de mamposterías

Las mamposterías se consideran elementos individuales, cuyas detecciones se encuentran ubicadas dentro de la matriz M.

$$M = \begin{bmatrix} x^{(1)}_M & y^{(1)}_M & w^{(1)}_M & h^{(1)}_M \\ \vdots & \vdots & \vdots & \vdots \\ x^{(m)}_M & y^{(m)}_M & w^{(m)}_M & h^{(m)}_M \end{bmatrix} \tag{84}$$

Al igual que los elementos anteriores, las mamposterías se diferencian en horizontales y verticales.

$$Mh = \begin{bmatrix} x_{Mh}^{(1)} & y_{Mh}^{(1)} & w_{Mh}^{(1)} \\ \vdots & \vdots & \vdots \\ x_{Mh}^{(m)} & y_{Mh}^{(m)} & w_{Mh}^{(m)} \end{bmatrix} \quad (85)$$

$$Mv = \begin{bmatrix} x_{Mv}^{(1)} & y_{Mv}^{(1)} & h_{Mv}^{(1)} \\ \vdots & \vdots & \vdots \\ x_{Mv}^{(m)} & y_{Mv}^{(m)} & h_{Mv}^{(m)} \end{bmatrix} \quad (86)$$

Los ejes de las cotas de la matriz Eh''' determinan las posibles posiciones de las mamposterías horizontales dentro del espacio real de dibujo y éstas se encuentran limitadas por los ejes de las cotas de la matriz Ev'''. Análogamente, los ejes de las cotas de la matriz Ev''' determinan las posibles posiciones de las mamposterías verticales y éstas se encuentran limitadas por los ejes de las cotas de la matriz Eh''.

Para determinar a cuáles de estas posiciones le corresponde una mampostería, se calcula la distancia del centro de las mamposterías al centro de cada uno de estos espacios y se verifica, para cada uno, si alguna de estas distancias es menor a un valor Δ_M. Esto se traduce en identificar los espacios dentro del dibujo donde se encuentre cercana alguna de las mamposterías, comparando los ejes horizontales con las mamposterías horizontales, y los ejes verticales con las mamposterías verticales.

Para facilitar esta verificación a la hora de programar, se preparan vectores donde se ubican las coordenadas de los centros de cada espacio. Estos vectores se definen como:

$$Uh_i = \begin{bmatrix} x_{Uh}^{(i)} & y_{Uh}^{(i)} \end{bmatrix} = \begin{bmatrix} e_{Ev'''}^{(i)} + \dfrac{e'^{(i)}_{Ev'''}}{2} & e_{Eh'''}^{(j)} \end{bmatrix} \quad (87)$$

$$Uv_i = \begin{bmatrix} x_{Uv}^{(i)} & y_{Uv}^{(i)} \end{bmatrix} = \begin{bmatrix} e_{Ev'''}^{(j)} & e_{Eh'''}^{(i)} + \dfrac{e'^{(i)}_{Eh'''}}{2} \end{bmatrix} \quad (88)$$

Donde los superíndices i y j representan el número de fila de las matrices Eh''' y Ev''', respectivamente.

Los valores de dh y dv para mamposterías horizontales y verticales, respectivamente, se definen como:

$$dh_{i,j} = \sqrt{\left(vh^{(i)} - x_{Mh}^{(j)}\right)^2 + \left(hh^{(i)} - y_{Mh}^{(j)}\right)^2} \quad (89)$$

$$dv_{i,j} = \sqrt{\left(vv^{(i)} - x_{Mv}^{(j)}\right)^2 + \left(hv^{(i)} - y_{Mv}^{(j)}\right)^2} \quad (90)$$

Y como se ha dicho, se debe cumplir que:

$$dh_{i,j} < \Delta_M \quad (91)$$

$$dv_{i,j} < \Delta_M \quad (92)$$

Donde Δ_M es un valor arbitrario. Para el presente trabajo se utilizó $\Delta_M = min\ (w_C^{(1)}, h_C^{(1)})$, con buenos resultados.

Las filas de las matrices Uh y Uv cuyas coordenadas no se encuentran cercanas ninguna mampostería, se eliminan de estas matrices.

3.5.5 Escala real de las mamposterías

Una vez determinado el par de ejes que determinan la ubicación de cada mampostería dentro de la imagen, se procede a transformar estos ejes a sus homólogos dentro del espacio real de dibujo:

$$Eh'''_m \rightarrow Ehr'''_m \quad (93)$$

$$Ev'''_m \rightarrow Evr'''_m \quad (94)$$

Los valores de las matrices Uh y Uv indican las coordenadas donde se ubican las mamposterías dentro de la imagen. La siguiente tarea será determinar estas coordenadas dentro del espacio real de dibujo:

$$Uhr_i = \begin{bmatrix} x_{Uhr}^{(i)} & y_{Uhr}^{(i)} \end{bmatrix} = \begin{bmatrix} e_{Evr'''}^{(i)} + \frac{e'^{(i)}_{Evr'''}}{2} & e_{Ehr'''}^{(j)} \end{bmatrix} \quad (95)$$

$$Uvr_i = \begin{bmatrix} x_{Uvr}^{(i)} & y_{Uvr}^{(i)} \end{bmatrix} = \begin{bmatrix} e_{Evr'''}^{(j)} & e_{Ehr'''}^{(i)} + \frac{e'^{(i)}_{Ehr'''}}{2} \end{bmatrix} \quad (96)$$

Coordenadas y dimensiones reales de cada una de estas mamposterías.

Construimos la matriz Mrh donde ubicamos las coordenadas de las mamposterías horizontales y la matriz Mrv donde ubicamos las coordenadas de las mamposterías verticales dentro del espacio real de dibujo, así como la longitud y espesor de cada una. Cada una de las filas de dichas matrices se definen como:

$$Mrh_i = \begin{bmatrix} x_{Mrh}^{(i)} & y_{Mrh}^{(i)} & w_{Mrh}^{(i)} & h_{Mrh}^{(i)} \end{bmatrix} = \begin{bmatrix} x_{Uhr}^{(i)} & y_{Uhr}^{(i)} & Num_{Eh'''}^{(i)} & ep \end{bmatrix} \quad (97)$$

$$Mrv_i = \begin{bmatrix} x_{Mrv}^{(i)} & y_{Mrv}^{(i)} & w_{Mrv}^{(i)} & h_{Mrv}^{(i)} \end{bmatrix} = \begin{bmatrix} x_{Uvr}^{(i)} & y_{Uvr}^{(i)} & ep & Num_{Ev'''}^{(i)} \end{bmatrix} \quad (98)$$

Donde ep representa el espesor de la mampostería.

Cada una de estas mamposterías puede entenderse como un rectángulo de base w y altura h, cuyo centro se ubica en las coordenadas x y y, dentro del espacio real de dibujo.

3.5.6 Ubicación de aberturas

Los procesos para la ubicación de aberturas de puertas y ventanas mencionados a continuación, son solamente métodos aproximados, por lo que pueden ser mejorados con otras técnicas en posteriores investigaciones. Sin embargo, estos métodos funcionan

muy bien cuando se trata de la realización de cómputos métricos, por ejemplo, ya que la variación con los planos suele ser por la ubicación de las aberturas y no por sus dimensiones. Dicho esto, con los siguientes métodos se obtienen cómputos muy cercanos a la realidad, por lo que se optó por su utilización.

3.5.6.1 Ubicación de puertas

La matriz donde se encuentran las detecciones de las puertas, tiene la forma:

$$P = \begin{bmatrix} x_P^{(1)} & y_P^{(1)} & w_P^{(1)} & h_P^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_P^{(m)} & y_P^{(m)} & w_P^{(m)} & h_P^{(m)} \end{bmatrix} \quad (99)$$

Primeramente, para uso posterior en el cómputo, conviene etiquetar las puertas según su uso dentro del croquis. Para este proyecto diferenciamos dos tipos de puertas; puertas de acceso a sanitarios y puertas de acceso general. El proceso de identificación propuesto se realiza comparando la distancia entre las puertas y las detecciones de los símbolos de baño, designando como puerta de baño a todas aquellas que se encuentren más cerca de los símbolos de baño.

Las detecciones de baños se encuentran almacenadas en una matriz de la forma:

$$B = \begin{bmatrix} x_b^{(1)} & y_b^{(1)} & w_b^{(1)} & h_b^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_b^{(m)} & y_b^{(m)} & w_b^{(m)} & h_b^{(m)} \end{bmatrix} \quad (100)$$

Podemos asignar una etiqueta "b" para las puertas de acceso a los baños; y "g" para las puertas generales. Cada una de estas etiquetas pueden agregarse al final de cada fila de las puertas.

Se asume que cada baño posee una sola puerta de ingreso, por lo que la identificación de las puertas de baño responde a:

$$P_{i,5} = \text{“}b\text{”} \rightarrow d_{i,m} \in Dmin \quad (101)$$

$$P_{i,5} = \text{“}g\text{”} \rightarrow d_{i,m} \notin Dmin \quad (102)$$

Donde:

$$Dmin = \{min(D_0), min(D_1), \dots, min(D_i)\} \quad (103)$$

$$D_i = \{d_{i,0}, d_{i,1}, \dots, d_{i,m}\} \quad (104)$$

$$d_{i,m} = \sqrt{\left(x_B^{(i)} - x_P^{(m)}\right)^2 + \left(y_B^{(i)} - y_P^{(m)}\right)^2} \quad (105)$$

De esta manera, la matriz P queda determinada como sigue:

$$P = \begin{bmatrix} x_P^{(1)} & y_P^{(1)} & w_P^{(1)} & h_P^{(1)} & etiqueta_P^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_P^{(m)} & y_P^{(m)} & w_P^{(m)} & h_P^{(\ )} & etiqueta_P^{(m)} \end{bmatrix} \quad (106)$$

Etiquetada cada puerta, se procede a realizar la ubicación de cada una de estas dentro de la imagen. Se debe asignar a cada puerta el eje al que pertenece y su ubicación en la misma. Para cada puerta se debe encontrar el eje vertical más cercano en el eje x y el eje horizontal más cercano en el eje y, entendiendo que estas distancias se calculan desde el centro de cada detección de puerta. Para esto realizamos los cálculos con las matrices Eh''' y Ev'''.

La determinación del eje vertical m más cercano se realiza como sigue:

$$eje \equiv {Ev'''}_m \mid dx_{i,m} = \min(Dx_i) \quad (107)$$

Donde:

$$Dx_i = \{dx_{i,0}, dx_{i,1}, \dots, dx_{i,m}\} \quad (108)$$

$$dx_{i,m} = \left| x_P^{(i)} - e_{Ev'''}^{(m)} \right| \quad (109)$$

Una vez identificado el eje, se debe definir si la puerta se encuentra hacia la derecha o hacia la izquierda de este en de la imagen. Para esto se realiza el siguiente análisis:

$$Ubicación\ (C_{(i)}) = \begin{cases} Izquierda \quad si \quad e_{Ev'''}^{(m)} > x_P^{(i)} \\ Derecha \quad si \quad e_{Ev'''}^{(m)} < x_P^{(i)} \end{cases} \quad (110)$$

La determinación del eje horizontal m más cercano se realiza como sigue:

$$eje \equiv {Eh'''}_m \mid dy_{i,m} = \min(Dy_i) \quad (111)$$

Donde:

$$Dy_i = \{dy_{i,0}, dy_{i,1}, \dots, dy_{i,m}\} \quad (112)$$

$$dy_{i,m} = \left| y_P^{(i)} - e_{Eh'''}^{(m)} \right| \quad (113)$$

Identificado el eje, se debe definir si la puerta se encuentra por encima o por debajo de este en la imagen. Para esto se realiza el siguiente análisis:

$$Ubicación\ (C_{(i)}) = \begin{cases} Superior \quad si \quad e_{Eh'''}^{(m)} > y_P^{(i)} \\ Inferior \quad si \quad e_{Eh'''}^{(m)} < y_P^{(i)} \end{cases} \quad (114)$$

De esta manera queda definido para cada puerta, uno de los cuadrantes formados por los ejes considerados cercanos. Resta determinar el eje al que la puerta pertenece, considerado el más cercano entre ambos:

$$eje \equiv \begin{cases} {Eh'''}_m \ si \ dx_{i,m} > dy_{i,m} \\ {Ev'''}_m \ si \ dx_{i,m} < dy_{i,m} \end{cases} \quad (115)$$

Una vez determinado el par de ejes cercanos de cada puerta dentro de la imagen, se procede a transformar estos ejes a sus homólogos dentro del espacio real de dibujo:

$$Eh'''_m \rightarrow Ehr'''_m \quad (116)$$

$$Ev'''_m \rightarrow Evr'''_m \quad (117)$$

Definimos la matriz Pr, donde cada fila representa una puerta dentro del espacio real de dibujo:

$$Pr_i = \begin{bmatrix} x_{Pr}^{(i)} & y_{Pr}^{(i)} & dim \end{bmatrix} \quad (118)$$

Donde dim es el ancho de la puerta, que puede variar entre dos valores según sea de acceso a un baño o de uso general. $x_{Pr}^{(i)}$ y $y_{Pr}^{(i)}$ toman los valores siguientes de acuerdo al eje y cuadrante al que pertenecen:

Si la puerta se ubica sobre el eje horizontal y la descripción que define la ubicación de la puerta es $Derecha$:

$$x_{Pr}^{(i)} = e_{Evr'''}^{(m)} + \frac{ep}{2} + em \quad (119)$$

$$y_{Pr}^{(i)} = e_{Ehr'''}^{(m)} - \frac{ep}{2} \quad (120)$$

Donde el valor de em representa la distancia arbitraria de la puerta con la pared perpendicular más cercana.

Si la puerta se ubica sobre el eje horizontal y la descripción que define la ubicación de la puerta es $Izquierda$:

$$x_{Pr}^{(i)} = e_{Evr'''}^{(m)} - \frac{ep}{2} - em - dim \quad (121)$$

$$y_{Pr}^{(i)} = e_{Ehr'''}^{(m)} - \frac{ep}{2} \quad (122)$$

Si la puerta se ubica sobre el eje vertical y la descripción que define la ubicación de la puerta es $Inferior$:

$$x_{Pr}^{(i)} = e_{Evr'''}^{(m)} - \frac{ep}{2} \quad (123)$$

$$y_{Pr}^{(i)} = e_{Ehr'''}^{(m)} + \frac{ep}{2} + em \quad (124)$$

Si la puerta se ubica sobre el eje vertical y la descripción que define la ubicación de la puerta es $Superior$:

$$x_{Pr}^{(i)} = e_{Evr'''}^{(m)} - \frac{ep}{2} \quad (125)$$

$$y_{Pr}^{(i)} = e_{Ehr'''}^{(m)} - \frac{ep}{2} - em - dim \quad (126)$$

3.5.6.2 Ubicación de ventanas

Las ventanas deben dibujarse con proporciones que las diferencien unas de otras dentro de la imagen. Así, se entiende que una ventana es igual a otra dentro de espacio real de dibujo si sus proporciones son similares dentro de la imagen, y dos ventanas son de diferentes dimensiones dentro del espacio real de dibujo en caso contrario.

La matriz donde se encuentran las detecciones de las ventanas, tiene la forma:

$$V = \begin{bmatrix} x_V^{(1)} & y_V^{(1)} & w_V^{(1)} & h_V^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_V^{(m)} & y_V^{(m)} & w_V^{(m)} & h_V^{(m)} \end{bmatrix} \quad (127)$$

La primera tarea propuesta a realizar con las ventanas detectadas es determinar la dimensión en planta de cada una. Estas medidas son colocadas en una lista por el usuario como sigue:

$$Vdim = [Vdim_1 \quad Vdim_2 \quad \dots \quad Vdim_n] \tag{128}$$

Donde cada uno de los valores $Vdim_n$ es una medida arbitraria de ventana existente dentro del plano.

Podemos definir un factor de escala para asignarle un valor de $Vdim$ a cada ventana dentro de la imagen. Este factor de escala se define como:

$$Vesc = \frac{\max(Vdim)}{\max(T)} \tag{129}$$

Donde:

$$T = \{w_V^{(1)}; h_V^{(1)}; \dots; w_V^{(m)}; h_V^{(m)}\} \tag{130}$$

El siguiente paso es determinar la orientación de cada ventana:

$$Orientación\ (V_{(i)}) = \begin{cases} Horizontal & si\ \ w_V^{(i)} > h_V^{(i)} \\ Vertical & si\ \ w_V^{(i)} < h_V^{(i)} \end{cases} \tag{131}$$

Teniendo el factor de escala y la orientación definidos, podemos asignar el valor de $Vdim$ más cercano a cada ventana con dimensión escalada, de la siguiente manera:

$$d_{Vr}^{(i)} = Vdim_n \mid min\ (D_i) = Vdim_n - max\left(w_V^{(i)}; h_V^{(i)}\right) Vesc \tag{132}$$

Donde:

$$D_i = \{Vdim_1 - max\left(w_V^{(i)}; h_V^{(i)}\right) Vesc; \dots; Vdim_n - max\left(w_V^{(i)}; h_V^{(i)}\right) Vesc\} \quad (133)$$

Las ubicaciones de las ventanas se determinan asignando el eje vertical más cercano a las ventanas verticales en sentido de las abscisas y el eje horizontal más cercano a las ventanas horizontales en sentido de las ordenadas.

La determinación del eje vertical m más cercano se realiza como sigue:

$$eje \equiv Ev'''_m \mid dx_{i,m} = \min(Dx_i) \quad (134)$$

Donde:

$$Dx_i = \{dx_{i,0}, dx_{i,1}, \dots, dx_{i,m}\} \quad (135)$$

$$dx_{i,m} = \left|x_V^{(i)} - e_{Ev'''}^{(m)}\right| \quad (136)$$

La determinación del eje horizontal m más cercano se realiza como sigue:

$$eje \equiv Eh'''_m \mid dy_{i,m} = \min(Dy_i) \quad (137)$$

Donde:

$$Dy_i = \{dy_{i,0}, dy_{i,1}, \dots, dy_{i,m}\} \quad (138)$$

$$dy_{i,m} = \left|y_V^{(i)} - e_{Eh'''}^{(m)}\right| \quad (139)$$

Además de esto, para cada ventana horizontal se debe encontrar un vector $Eh'''_{(m)}$ cuyos valores cumplan:

$$e_{Ev'''}^{(m)} > x_V^{(i)} > e'^{(m)}_{Ev'''} \quad (140)$$

Y para cada ventana vertical un vector $Ev'''_{(m)}$ cuyos valores cumplan:

$$e_{Eh'''}^{(m)} > y_V^{(i)} > e'^{(m)}_{Eh'''} \quad (141)$$

Una vez determinado el par de ejes que determinan la ubicación de cada ventana dentro de la imagen, se procede a transformar estos ejes a sus homólogos dentro del espacio real de dibujo:

$$Eh'''_m \rightarrow Ehr'''_m \quad (142)$$

$$Ev'''_m \rightarrow Evr'''_m \quad (143)$$

Se determina la matriz Vr, donde cada fila representa una ventana dentro del espacio real de dibujo.

Cada fila de la matriz Vr se define como:

$$Vr_i = \begin{bmatrix} x_{Vr}^{(i)} & y_{Vr}^{(i)} & d_{Vr}^{(i)} \end{bmatrix} \quad (144)$$

Donde $x_{Vr}^{(i)}$ y $y_{Vr}^{(i)}$ toman los siguientes valores para el caso de las ventanas verticales:

$$x_{Vr}^{(i)} = e_{Evr'''}^{(m)} \quad (145)$$

$$y_{Pr}^{(i)} = e_{Ehr'''}^{(m)} + \frac{e'^{(i)}_{Ehr'''} - e_{Ehr'''}^{(m)}}{2} \quad (146)$$

Y para el caso de las ventanas horizontales:

$$x_{Vr}^{(i)} = e_{Evr'''}^{(m)} + \frac{e'^{(i)}_{Evr'''} - e_{Evr'''}^{(m)}}{2} \quad (147)$$

$$y_{Pr}^{(i)} = e'^{(m)}_{Evr'''} \quad (148)$$

3.5.7 Asignación de losas a escala

Una losa se define como el espacio determinado por un par de vectores Eh'''_m y Ev'''_m que contiene en su interior y cerca de su centro, un símbolo de losa.

La matriz donde se encuentran las detecciones de las losas, tiene la forma:

$$L = \begin{bmatrix} x_L^{(1)} & y_L^{(1)} & w_L^{(1)} & h_L^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_L^{(m)} & y_L^{(m)} & w_L^{(m)} & h_L^{(m)} \end{bmatrix} \quad (149)$$

El vector ${Ev'''}_m$ que define los extremos izquierdo y derecho de la losa es aquel que cumple con la siguiente condición:

$$dx_{i,m} = \min(Dx_i) \quad (150)$$

Donde:

$$Dx_i = \{dx_{i,0}, dx_{i,1}, \dots, dx_{i,m}\} \quad (151)$$

$$dx_{i,m} = \left| x_L^{(i)} - e_{Ev'''}^{(m)} - \frac{{e'}_{Ev'''}^{(m)}}{2} \right| \quad (152)$$

El vector ${Eh'''}_m$ que define los extremos superior e inferior de la losa es aquel que cumple con la siguiente condición:

$$dy_{i,m} = \min(Dy_i) \quad (153)$$

Donde:

$$Dy_i = \{dy_{i,0}, dy_{i,1}, \dots, dy_{i,m}\} \quad (154)$$

$$dy_{i,m} = \left| y_L^{(i)} - e_{Eh'''}^{(m)} - \frac{{e'}_{Eh'''}^{(m)}}{2} \right| \quad (155)$$

Una vez determinado el par de ejes que determinan la ubicación de cada ventana dentro de la imagen, se procede a transformar estos ejes a sus homólogos dentro del espacio real de dibujo:

$$Eh'''_m \rightarrow Ehr'''_m \quad (156)$$

$$Ev'''_m \rightarrow Evr'''_m \quad (157)$$

Así, definimos los vectores de la matriz Lr como sigue:

$$Lr_i = \begin{bmatrix} x_{Lr}^{(i)} & y_{Lr}^{(i)} & w_{Lr}^{(i)} & h_{Lr}^{(i)} \end{bmatrix} \quad (158)$$

Donde:

$$x_{Lr}^{(i)} = e'^{(m)}_{Evr'''} - \frac{e_{Evr'''}^{(m)}}{2} + \frac{ep}{2} \quad (159)$$

$$y_{Lr}^{(i)} = e'^{(m)}_{Ehr'''} - \frac{e_{Ehr'''}^{(m)}}{2} + \frac{ep}{2} \quad (160)$$

$$w_{Lr}^{(m)} = Cr_{Ev'''}^{(i)} + ep \quad (161)$$

$$h_{Lr}^{(m)} = Cr_{Eh'''}^{(i)} + ep \quad (162)$$

El siguiente paso es ordenar de manera ascendente los vectores Lr_i según la altura de sus centros dentro del espacio real de dibujo, es decir, según el valor $y_{Lr}^{(i)}$ de cada vector, y agrupar aquellos que tengan la misma altura. Luego, cada uno estos grupos debe ordenarse de manera ascendente según el valor $x_{Lr}^{(i)}$.

El último paso es determinar las condiciones de bordes de cada losa, es decir, determinar las uniones que tiene cada losa en sus bordes. Existen dos posibilidades: que el borde de la losa limite con el borde de otra losa; y que el borde de la losa no limite con ninguna otra. Esta determinación se realiza para cada borde de cada losa como sigue:

A cada losa Lr_i se le asigna un vector de conexiones $Lconect_i$ que se define como:

$$Lconect_i = [ar^{(i)} \quad iz^{(i)} \quad ab^{(i)} \quad de^{(i)}] \quad (163)$$

Donde los valores $ar^{(i)}$, $iz^{(i)}$, $ab^{(i)}$, y $de^{(i)}$ pueden tomar el valor de la losa con la que se encuenta limitando y -1 en caso de no tener conexión con ninguna losa, en su extremo superior, izquierdo, inferior y derecho respectivamente.

Una losa Lr_i limita en su borde superior con una losa Lr_j si se cumplen las siguientes condiciones:

$$y_{Lr}^{(i)} - \frac{h_{Lr}^{(i)}}{2} = y_{Lr}^{(j)} + \frac{h_{Lr}^{(j)}}{2} \quad (164)$$

$$x_{Lr}^{(j)} + \frac{w_{Lr}^{(j)}}{2} > x_{Lr}^{(i)} > x_{Lr}^{(j)} - \frac{w_{Lr}^{(j)}}{2} \quad (165)$$

En caso de cumplirse con estas condiciones, $ar^{(i)}$ toma el valor j, en caso contrario toma el valor de -1.

Una losa Lr_i limita en su borde izquierdo con una losa Lr_j si se cumplen las siguientes condiciones:

$$x_{Lr}^{(i)} - \frac{w_{Lr}^{(i)}}{2} = x_{Lr}^{(j)} + \frac{w_{Lr}^{(j)}}{2} \quad (166)$$

$$y_{Lr}^{(j)} + \frac{h_{Lr}^{(j)}}{2} > y_{Lr}^{(i)} > y_{Lr}^{(j)} - \frac{h_{Lr}^{(j)}}{2} \quad (167)$$

En caso de cumplirse con estas condiciones, $iz^{(i)}$ toma el valor j, en caso contrario toma el valor de -1.

Una losa Lr_i limita en su borde inferior con una losa Lr_j si se cumplen las siguientes condiciones:

$$y_{Lr}^{(i)} + \frac{h_{Lr}^{(i)}}{2} = y_{Lr}^{(j)} - \frac{h_{Lr}^{(j)}}{2} \quad (168)$$

$$x_{Lr}^{(j)} + \frac{w_{Lr}^{(j)}}{2} > x_{Lr}^{(i)} > x_{Lr}^{(j)} - \frac{w_{Lr}^{(j)}}{2} \quad (169)$$

En caso de cumplirse con estas condiciones, $ab^{(i)}$ toma el valor j, en caso contrario toma el valor de -1.

Una losa Lr_i limita en su borde derecho con una losa Lr_j si se cumplen las siguientes condiciones:

$$x_{Lr}^{(i)} + \frac{w_{Lr}^{(i)}}{2} = x_{Lr}^{(j)} - \frac{w_{Lr}^{(j)}}{2} \quad (170)$$

$$y_{Lr}^{(j)} + \frac{h_{Lr}^{(j)}}{2} > y_{Lr}^{(i)} > y_{Lr}^{(j)} - \frac{h_{Lr}^{(j)}}{2} \quad (171)$$

En caso de cumplirse con estas condiciones, $de^{(i)}$ toma el valor j, en caso contrario toma el valor de -1.

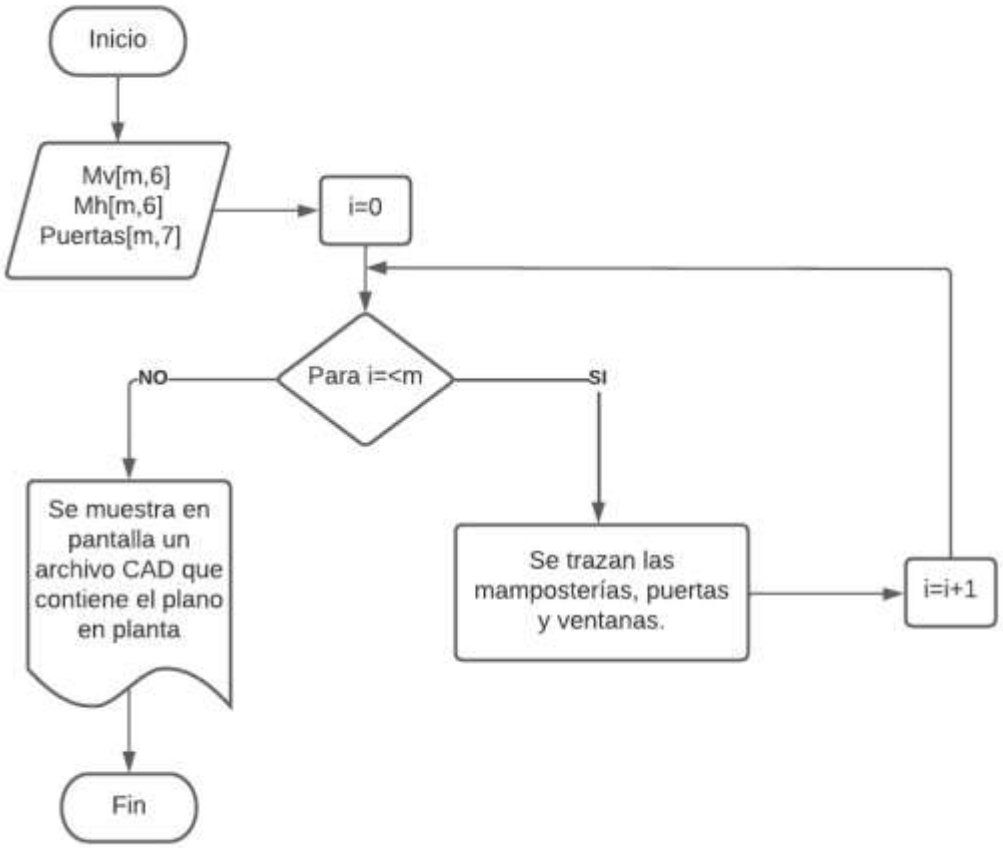
$$Lr_i = \begin{bmatrix} x_{Lr}^{(i)} & y_{Lr}^{(i)} & w_{Lr}^{(i)} & h_{Lr}^{(i)} \end{bmatrix} \quad (172)$$

3.6 Algoritmo para obtención de planos CAD

El algoritmo para obtención de planos CAD puede reflejar básicamente su funcionamiento en el siguiente diagrama de flujo. Posteriormente, puede observarse una explicación más detallada del mismo.

Figura 32

Diagrama de flujo de Algoritmo para obtención de planos CAD.



Para proceder a la ejecución del algoritmo, primeramente se debe de tener el entorno de dibujo AutoCAD abierto. El algoritmo parte con la recepción de datos del programa de reconstrucción. Comienza recorriendo la matriz $Puertas$ utilizando un bucle

`for` y una sentencia `if/else`. En los mismos se verifica que se cumpla la condición, si la base de la puerta es mayor a la altura de la misma, La fila de la matriz analizada se almacena en una matriz llamada Ph que representa a las puertas horizontales. En caso contrario se almacenan en una matriz llamada Pv que representa a las puertas verticales. Estas matrices tienen una dimensión de $mx7$.

$$Ph = \begin{bmatrix} cixsi_{Ph}^{(1)} & ciysi_{Ph}^{(1)} & mp_{Ph}^{(1)} & mm_{Ph}^{(1)} & "h" & ex_{Ph}^{(1)} & ey_{Ph}^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ cixsi_{Ph}^{(m)} & ciysi_{Ph}^{(m)} & mp_{Ph}^{(m)} & mm_{Ph}^{(m)} & "h" & ex_{Ph}^{(m)} & ey_{Ph}^{(m)} \end{bmatrix} \quad (173)$$

$$Pv = \begin{bmatrix} cixsi_{Pv}^{(1)} & ciysi_{Pv}^{(1)} & mp_{Pv}^{(1)} & mm_{Pv}^{(1)} & "v" & ex_{Pv}^{(1)} & ey_{Pv}^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ cixsi_{Pv}^{(m)} & ciysi_{Pv}^{(m)} & mp_{Pv}^{(m)} & mm_{Pv}^{(m)} & "v" & ex_{Pv}^{(m)} & ey_{Pv}^{(m)} \end{bmatrix} \quad (174)$$

Donde cada columna representan en el orden de izquierda a derecha lo siguiente:

- La coordenada de inicio de la puerta en x.
- La coordenada de inicio de la puerta en y.
- La medida de la puerta.
- El espesor de la mampostería donde están los ejes más cercanos a la puerta.
- El identificador de la puerta horizontal "h" o vertical "v".
- La posición sobre el eje x en los cuadrantes formados por los ejes de las mamposterías más cercanas a la puerta, pudiendo ser izquierda o derecha.
- La posición sobre el eje y en los cuadrantes formados por los ejes de las mamposterías más cercanas a la puerta, pudiendo ser superior o inferior.

3.6.1 Trazado de mamposterías y aberturas de puertas

El algoritmo procederá a crear las capas de mamposterías, puertas y ventanas. Seguidamente se crea una variable k y le asigna el valor cero. Se abre un bucle `for` con

sentencias `if`, que recorren las matrices Mv, Mh, Pv y Ph. Si se cumple la condición de que las coordenadas en x de las puertas verticales coinciden con las coordenadas en x de las mamposterías verticales o si se cumple la condición de que las coordenadas en y de las puertas horizontales coinciden con las coordenadas en y de las mamposterías horizontales. Se procede a analizar si la ubicación de la coordenada del punto de inicio de la puerta analizada es mayor a la coordenada del punto de inicio de la mampostería analizada y la ubicación de la coordenada del punto final de la puerta analizada es menor a la coordenada del punto final de la mampostería analizada, el algoritmo asume que la puerta se encuentra dentro de esa mampostería y k toma el valor de uno. Entonces dibuja polilíneas de acuerdo a la ubicación de las coordenadas de los puntos de las mamposterías y las puertas como también el espesor de la mampostería, tomando la ubicación que les corresponde en el plano de dibujo.

En el caso contrario, si el algoritmo no encuentra puertas en la mampostería, k sigue siendo igual a cero, entonces se dibuja la mampostería de acuerdo a la ubicación de los puntos de inicio, fin y su correspondiente espesor.

3.6.2 Trazado de puertas

Se abre un bucle `for` con sentencias `if/else`, para analizar las puertas horizontales o verticales de las matrices Pv y Ph, se verifica si la posición sobre el eje en los cuadrantes formados por los ejes de las mamposterías más cercanas a la puerta, pudiendo ser izquierda o derecha para las puertas horizontales y superior o inferior para las puertas verticales.

De acuerdo a la condición que se cumpla, el algoritmo procederá a trazar el dibujo de una puerta abierta, ubicándola en sus coordenadas correspondientes en el entorno de dibujo.

3.6.3 Trazado de ventanas

Se abre un bucle `for` con sentencias `if`, que recorren las matrices de mamposterías verticales Mv y horizontales Mh, las matrices de las ventanas verticales Vv y horizontales Vh. Si se cumple la condición de que las coordenadas en x de las ventanas verticales coinciden con las coordenadas en x de las mamposterías verticales o si se cumple la condición de que las coordenadas en y de las ventanas horizontales coinciden con las coordenadas en y de las mamposterías horizontales. Se procede a analizar si la ubicación de la coordenada del punto de inicio de la ventana analizada es mayor a la coordenada del punto de inicio de la mampostería analizada y la ubicación de la coordenada del punto final de la ventana analizada es menor a la coordenada del punto final de la mampostería analizada, el algoritmo asume que la ventana se encuentra dentro de esa mampostería. Entonces dibuja polilíneas que representan a una ventana con su marco donde su centro corresponde al centro de la mampostería donde se encuentra.

3.6.4 Cotas

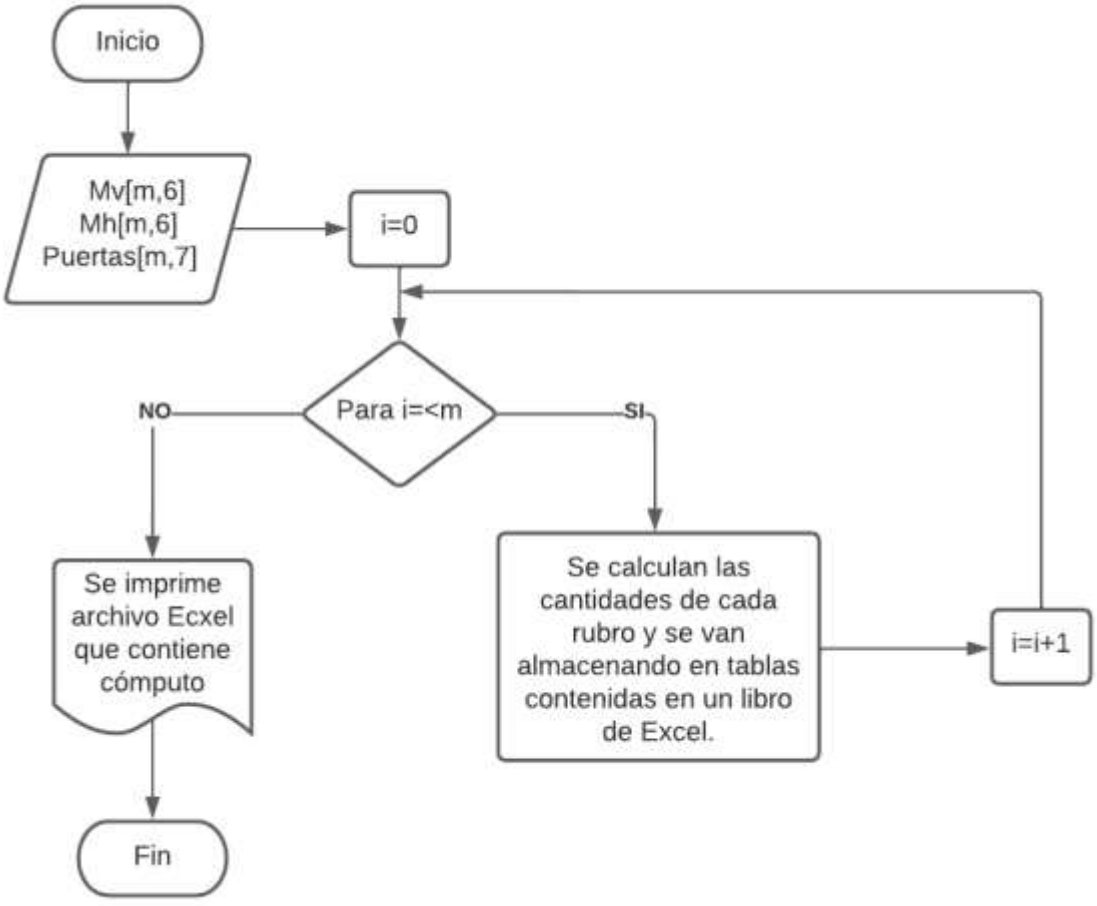
A partir de las matrices de las mamposterías, dentro de un bucle `for` se recorren las coordenadas de inicio como también final de cada mampostería y con ellas se trazan las cotas con las medidas internas de cada habitación que compone al plano. Una vez concluido este último proceso, se puede apreciar el plano en planta del boceto arquitectónico inicial convertido en un archivo de formato CAD.

3.7 Algoritmo para obtención de Cómputos

El algoritmo de Cómputos puede reflejar básicamente su funcionamiento en el siguiente diagrama de flujo. Posteriormente, puede observarse una explicación más detallada del mismo.

Figura 33

Diagrama de flujo de Algoritmo de Cómputos.



3.7.1 Cálculo de perímetros

El algoritmo recibe los datos del programa de reconstrucción, procede a leer el espesor ep y las matrices Mv y Mh. Explicadas anteriormente en el apartado del algoritmo de reconstrucción.

En bucles `for` con sentencias `if` van recorriendo y analizando si uno de los lados de la mampostería vertical u horizontal se encuentra afuera, las variables h y b van almacenando los valores de las longitudes correspondientes a las paredes verticales y horizontales respectivamente. Estas variables se almacenan en una variable que representa al perímetro medio o perímetro entre ejes de la construcción, llamada Pm. En base a la misma variable se puede obtener el perímetro total exterior de la construcción. Definida con la siguiente ecuación.

$$P = Pm + 4ep \tag{175}$$

3.7.2 Cálculo de áreas

Se inicializan bucles `for` con sentencias `if/else` A partir de las matrices Mv y Mh, se hace un recorrido de izquierda a derecha y de arriba hacia abajo respectivamente. Para Mv, si el lado derecho de la mampostería se encuentra en el interior. O en caso contrario si el lado derecho y el izquierdo están afuera, el algoritmo procede a calcular valores y cargarlos en la siguiente matriz.

$$ai = \begin{bmatrix} cmayd^{(1)} & cma^{(1)} & dmayd^{(1)} & lma^{(1)} & Amayd^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ cmayd^{(m)} & cma^{(m)} & dmayd^{(m)} & lma^{(m)} & Amayd^{(m)} \end{bmatrix} \tag{176}$$

Donde cada columna representan en el orden de izquierda a derecha lo siguiente:

- Centro entre la mampostería analizada y la mampostería que se encuentra a su derecha.
- Centro de la mampostería analizada.
- Distancia entre la mampostería analizada y la mampostería que se encuentra a su derecha.
- La longitud de la mampostería.
- Área entre la mampostería analizada y la mampostería que se encuentra a su derecha.
- Los valores de esta última columna se suman en la variable Ai, que representa el área interna de la construcción que se obtuvo en el recorrido de izquierda a derecha.

Para Mh, si el lado de abajo de la mampostería se encuentra en el interior. O en caso contrario si el lado de abajo y el de arriba están afuera, el algoritmo procede a calcular valores y cargarlos en la siguiente matriz.

$$aa = \begin{bmatrix} cmaya^{(1)} & cma^{(1)} & dmaya^{(1)} & lma^{(1)} & Amaya^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ cmaya^{(m)} & cma^{(m)} & dmaya^{(m)} & lma^{(m)} & Amaya^{(m)} \end{bmatrix} \quad (177)$$

Donde cada columna representan en el orden de arriba hacia abajo lo siguiente:

- Centro entre la mampostería analizada y la mampostería que se encuentra hacia abajo.
- Centro de la mampostería analizada.
- Distancia entre la mampostería analizada y la mampostería que se encuentra hacia abajo.
- La longitud de la mampostería.

- Área entre la mampostería analizada y la mampostería que se encuentra hacia abajo.

Los valores de esta última columna se suman en la variable Aa, que representa el área interna de la construcción que se obtuvo en el recorrido de arriba hacia abajo.

Seguidamente se calcula el área de borde con la siguiente ecuación.

$$Ab = (Pm + 2ep) * ep/2 \quad (178)$$

Y se calcula el área total con la siguiente ecuación.

$$A = máx(Ai, Aa) + Ab \quad (179)$$

Posteriormente con una sentencia $\texttt{if/else}$ se verifica lo siguiente, si Ai es mayor a Aa. La matriz a tomará los valores de ai, en caso contrario, la matriz a tomará los valores de aa.

Finalmente se abre un bucle $\texttt{for}$, que va recorriendo las filas de la matriz a, restándole el valor de ep a los valores de la columna 3 y 4 y reasignando dicho valor a la matriz.

3.7.3 Asignación de baños

Se abren 2 bucles $\texttt{for}$ anidados, que recorren las matrices $baños$ y a, se obtiene la distancia entre la ubicación del símbolo de baño y el centro del área analizada, al obtenerse la menor distancia, se le asigna "b" a la posición donde se encuentra el baño.

3.7.4 Longitud y sección transversal de mamposterías

Con bucles $\texttt{for}$ se recorren las matrices de Mv y Mh, se analizan si son mamposterías interiores, de ser el caso se resta a la longitud el valor de ep, para obtener

el perímetro de cada pared interior y se le suma al perímetro medio para obtener la longitud de mampostería total Lm. Seguidamente, para calcular el área transversal Amp se multiplica Lm por ep.

3.7.5 Cantidad de puertas

La cantidad de puertas para los baños es igual a la cantidad de baños cpb, la cantidad de las demás puertas es igual a la cantidad de puestas menos la de los baños cp.

3.7.6 Área de mamposterías

El área de las mamposterías Am, está dada por la siguiente ecuación

$$Am = Lm * altura - cp * (puertadim + 0{,}1) * 2{,}1 - cbp * (puertadim + 0{,}1) * 2{,}1 \quad (180)$$

3.7.7 Área de pisos

El área de pisos Ap, está dada por la siguiente ecuación

$$Ap = A - Amp \quad (181)$$

3.7.8 Áreas de mamposterías externas e internas

Se abre un bucle for y se recorren las matrices Mv y Mh. Se analizan si las mamposterías están en el interior sus longitudes se les resta ep, y el resultado se van sumando en la variable Ri. Luego se realiza dos reasignaciones. La primera es $Ri = Ri * altura$ y la segunda, con la que se obtiene la cantidad de áreas interiores, dada por la siguiente ecuación.

$$Ri = Ri - cp * 2{,}1 * puertadim - cbp * 2{,}1 * puertadim \quad (182)$$

Seguidamente para obtener la cantidad de áreas exteriores, se realiza la siguiente operación $Re = P * altura$ y luego una reasignación dada por la siguiente ecuación.

$$Re = Re - cp * 2{,}1 * puertadim - \ cbp * 2{,}1 * puertadim \quad (183)$$

3.7.9 Área de azulejos

Con un bucle `for` y una sentencia `if`. Se recorre la matriz a, si se encuentra el identificador de baño "b", se obtiene el perímetro del baño se la multiplica por la altura estándar de azulejo y se le resta el área de la puerta la puerta. Esto se va acumulando en la variable azu.

3.7.10 Volumen de hormigón

En base a los puntos de inicio y fin de cada mampostería, asumiendo que cada mampostería puede tener hasta dos pilares, el cálculo preliminar de la cantidad de pilares consiste en duplicar la cantidad de mamposterías horizontales y restándole a esta cantidad el número de pilares compartidas por dos mamposterías horizontales. A partir de la obtención del número de pilares el algoritmo hace el producto de la cantidad con las alturas entre pisos y la sección transversal de los pilares, previamente definidas por el usuario.

Para el cálculo de volumen de vigas, el algoritmo toma de base la suma de las longitudes de las mamposterías, para luego multiplicarlas por la base y altura que el usuario optó con anterioridad.

Para el volumen de las losas se utiliza el área total de la construcción por el espesor optado por el usuario.

Cada una de las variables y matrices definidas anteriormente nos servirán para calcular los siguientes rubros:

- Limpieza y replanteo.
- Mamposterías.
- Aislaciones.
- Revoques interiores y exteriores.
- Pinturas.
- Contrapisos.
- Pisos.
- Zócalos.
- Revestimientos (Azulejos).
- Aberturas (Puertas y ventanas).
- Hormigón (Losas, vigas y pilares).

3.7.11 Exportación de resultados a planilla de Excel

El algoritmo procederá a cargar los valores obtenidos en una plantilla Excel por defecto. En caso de que el usuario, desee utilizar una plantilla que sea más de su agrado, el mismo debe indicar al algoritmo la ubicación de la plantilla de cómputo. Pero debe de tener en cuenta que los rubros deben de ubicarse en las celdas de la segunda columna de la hoja del libro Excel y el valor de las cantidades de dichos rubros en las celdas de la cuarta columna de la misma.

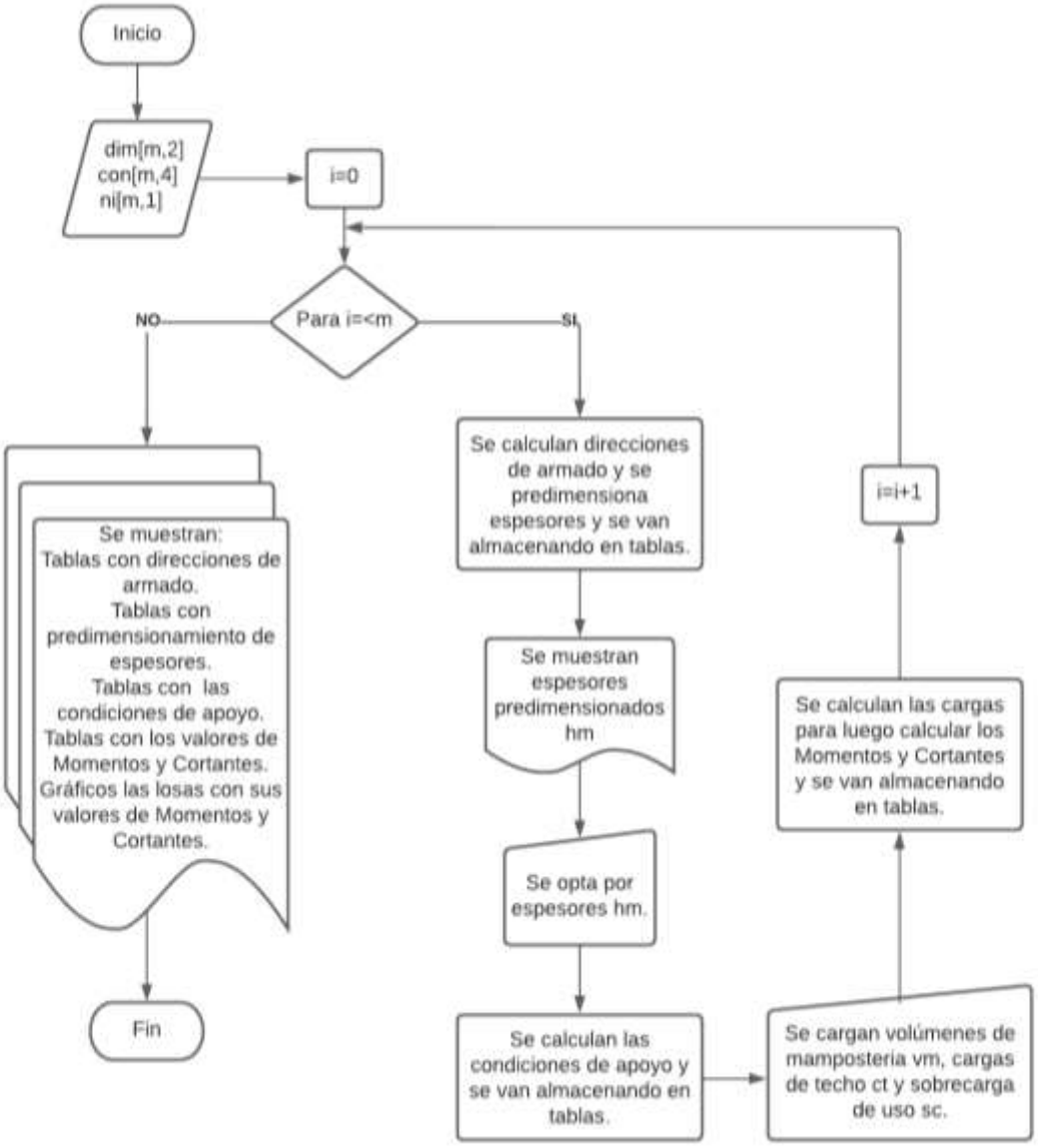
Una vez tenido en cuenta este parámetro, se abre un bucle `for` que va recorriendo las celdas de la cuarta columna y las va vaciando. Posteriormente se abre otro bucle `for` con sentencias if que van recorriendo las celdas de la segunda columna que corresponde a la de los rubros. Viendo si encuentra coincidencias entre rubros (El algoritmo tiene en cuenta la falta de mayúsculas y acentos), si encuentra coincidencias, este le asigna a la cuarta columna el valor de la cantidad que le corresponde a cada rubro.

3.8 Algoritmo de Cálculo de Solicitaciones en Losas

El algoritmo de predimensionado y cálculo de solicitaciones en losas puede reflejar básicamente su funcionamiento en el siguiente diagrama de flujo. Posteriormente, puede observarse una explicación más detallada del mismo.

Figura 34

Diagrama de flujo de Algoritmo de Losas.



3.8.1 Datos de entrada

El funcionamiento del algoritmo parte de la recepción de datos del programa de reconstrucción, los datos que reciben son las dimensiones en las direcciones de las coordenadas x e y. seguidamente de las conexiones con cada losa vecina, en caso de tenerlas.

Estos datos se almacenan en dos matrices, una llamada dim que posee una dimensión de $mx2$, donde m es el número de losas que posee la estructura.

Cada fila representa la numeración de la losa que se está analizando, partiendo desde el número 0 y cada columna representa yendo de izquierda a derecha representa las dimensiones en x y en y respectivamente.

$$dim = \begin{bmatrix} x_{dim}^{(1)} & y_{dim}^{(1)} \\ \vdots & \vdots \\ x_{dim}^{(m)} & y_{dim}^{(m)} \end{bmatrix} \tag{184}$$

La otra matriz es la matriz de conexiones entre losas, llamada con que tiene una dimensión de $mx4$. Cada fila representa la numeración de la losa que se está analizando, partiendo desde el número 0 y cada columna representa las conexiones de arriba, derecha, abajo e izquierda, recorriendo de izquierda a derecha. En esta matriz se almacenan los valores del número de losa con la que se conecta, por ejemplo, si se conecta con la losa 2 en el costado izquierdo, tomará el valor de 2 en la fila de la losa analizada y la columna que corresponde al lado izquierdo, si ocurre el caso de no estar conectada a ninguna losa, es decir, al encontrarse apoyada en la viga perimetral, la matriz tomará el valor de -1.

$$con = \begin{bmatrix} arr_{con}^{(1)} & der_{con}^{(1)} & aba_{con}^{(1)} & izq_{con}^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ arr_{con}^{(m)} & der_{con}^{(m)} & aba_{con}^{(m)} & izq_{con}^{(m)} \end{bmatrix} \quad (185)$$

Otro conjunto de datos que también recibe, son los niveles en que se encuentran cada losa, por ejemplo en el caso de que la red detecte un baño, la losa perteneciente al mismo, se tomará a un nivel inferior a las demás losas. Estos datos se almacenan en una matriz llamada ni de dimensión $mx1$, que tomará los valores de 1 en caso de ser una losa que está al mismo nivel que la mayoría de losas y el valor de 0 en caso de ser una losa que se encuentra en un nivel inferior o superior.

$$ni = \begin{bmatrix} ni^{(1)} \\ \vdots \\ ni^{(m)} \end{bmatrix} \quad (186)$$

3.8.2 Determinación de dirección de armado

Partiendo de los datos de entrada almacenadas en las matrices explicadas anteriormente, se crea un bucle `for`, que se encarga de recorrer las filas de la matriz que contiene las dimensiones de las losas, dentro del mismo bucle se ejecuta una sentencia if/`else`, que tiene la función de verificar la relación entre el máximo de las dimensiones en x o en y, con el mínimo de las dimensiones en x o en y. Si esta relación es menor o igual a dos, se obtiene que la losa es "Armada en dos direcciones", en caso contrario es "Armada en una dirección".

Estas definiciones se almacenan en una matriz de $mx1$, llamada d. Que a partir de la misma se imprimirá en forma de tabla donde aparecerá la dirección de armado con el correspondiente número de losa.

$$d = \begin{bmatrix} d^{(1)} \\ \vdots \\ d^{(m)} \end{bmatrix} \tag{187}$$

3.8.3 Predimensionamiento de espesor

Ahora, a partir de esta matriz d, se procede a predimensionar el espesor de la losa, este cálculo se realiza con una sentencia `if/else` dentro de un bucle `for`.

Teniendo en cuenta la dirección de armado de las losas, el algoritmo adopta el espesor de losa en base al Código [3]. Detallados en el capítulo anterior. Estos espesores se almacenan en una matriz de dimensión $mx1$, llamada hm.

$$hm = \begin{bmatrix} hm^{(1)} \\ \vdots \\ hm^{(m)} \end{bmatrix} \tag{188}$$

Si se opta por un espesor específico para cada losa, en la siguiente línea de código se puede modificar los valores de hm. De no ser el caso, el algoritmo sigue calculando con los valores obtenidos anteriormente.

3.8.4 Condiciones de apoyo

Una vez definidas las matrices anteriores, se analizan las condiciones de apoyo de cada losa, comenzando por leer las filas de la matriz con. Posteriormente se realiza una sentencia `if/else`, que en caso de ser solo una fila, lo indica que es una sola losa apoyada en sus bordes, Indistintamente si es en una losa armada en una o dos direcciones, toma como "Apoyada" en sus cuatro bordes.

En la situación que se tenga más losas, por ende más filas. Se abre un bucle `for`, que se encarga de recorrer las matrices con, dim, hm, y ni. Analizando con sentencias

`if/else`, cada una de las posibles condiciones de apoyo y definirlas de acuerdo a las consideraciones explicadas en el capítulo anterior.

Obtenidas las condiciones de apoyo, estas se almacenan en una matriz llamada $cemp$, que tiene una dimensión de $mx4$, donde cada columna representa las direcciones arriba, derecha, abajo e izquierda.

$$cemp = \begin{bmatrix} cemp_{arr}^{(1)} & cemp_{der}^{(1)} & cemp_{aba}^{(1)} & cemp_{izq}^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ cemp_{arr}^{(m)} & cemp_{der}^{(m)} & cemp_{aba}^{(m)} & cemp_{izq}^{(m)} \end{bmatrix} \quad (189)$$

A partir de esta matriz se genera una tabla que muestra el número de la losa y las condiciones de apoyo para cada lado de la misma.

3.8.5 Análisis de cargas

A diferencia de los pasos anteriores, para el análisis se presentan varias variables, es decir factores que en cada caso son distintos y que por ello no se puede generalizar, el usuario tendrá que definir y cargar ciertos valores, tales como los pesos específicos del hormigón γha y de la mampostería γm, peso por metro cuadrado de contrapiso pc, la sobrecarga según el tipo de uso que se le dará a la estructura sc y las matrices de dimensiones $mx1$ que componen al volumen de mampostería vm y el peso de techo pt.

$$vm = \begin{bmatrix} vm^{(1)} \\ \vdots \\ vm^{(m)} \end{bmatrix} \quad (190)$$

$$pt = \begin{bmatrix} pt^{(1)} \\ \vdots \\ pt^{(m)} \end{bmatrix} \quad (191)$$

A partir de los valores definidos, dentro de un bucle `for` con sentencias `if/else`, se proceden a calcular las cargas permanentes y cargas vivas de diseño de acuerdo al Código [3]. Las cargas se almacenan en las matrices cp y cv respectivamente con dimensiones $mx1$.

$$cp = \begin{bmatrix} cp^{(1)} \\ \vdots \\ cp^{(m)} \end{bmatrix} \tag{192}$$

$$cv = \begin{bmatrix} cv^{(1)} \\ \vdots \\ cv^{(m)} \end{bmatrix} \tag{193}$$

3.8.6 Cálculo de Momentos y Cortantes

Ya teniendo las cargas, el algoritmo abrirá un bucle `for` que recorrerá todas las matrices definidas anteriormente, que poseen números de filas correspondientes a la cantidad de losas que se encuentra en la estructura. Dentro del mismo bucle, estará una sentencia `if/else` que analizará la matriz d, para ver si la losa analizada está "Armada en dos direcciones" o "Armada en una dirección".

Si se encuentra en la categoría de "Armada en dos direcciones", se abrirá otro bucle `for`, que contendrá a la sentencia `if/else` la cual en conjunto con el bucle, tendrá la función de comparar cada vector correspondiente a las filas de la matriz $cemp$ con cada uno de los vectores correspondientes a las filas de una matriz llamada ca, esta matriz posee todas las posibles combinaciones de apoyos en los cuatro lados de una losa y cada fila es asignada a un caso particular. Si se da la coincidencia entre los vectores de las matrices se guardará el número de caso correspondiente en una variable. A continuación se abrirá otra sentencia `if/else` que evaluará al caso guardado en la

variable anterior con el número que corresponde a dicho caso. Esto ocurrirá hasta que coincida con el número de caso que corresponda a la losa analizada en ese momento. Cuando esto ocurra se abrirá otro bucle `for` en conjunto con una sentencia `if/else`, que se encargará de verificar la relación entre las dimensiones de las losas obtenidas de la matriz dim y con esa relación, recorrer la matriz correspondiente al caso. Por ejemplo si es el caso 2, recorrerá la matriz llamada $ca2$. Estas matrices contienen los coeficientes de las tablas de Marcus mencionadas en el capítulo anterior.

Continuando con el recorrido de la matriz para un caso particular. Dada la relación de las dimensiones en x e y, el algoritmo realizará interpolaciones para obtener el valor exacto de cada uno de los coeficientes y así obtener el valor de cada uno de los momentos y cortantes actuantes en la losa.

En caso contrario si se encuentra en la categoría de "Armada en una dirección", se realiza sentencias de `if/else` anidados a fin de comparar con cada caso posible y obtener los valores de momentos y cortantes para cada losa. Estos valores se obtienen en base a los coeficientes del Código [3], mencionadas en el capítulo anterior.

Una vez recorridas todas las losas y haber obtenidos todos los valores de momentos y cortantes, estas se almacenan en una matriz de dimensión $mx8$ llamada mom.

$$mom = \begin{bmatrix} MyLD(-)^{(1)} & MxLD(-)^{(1)} & MyD(+)^{(1)} & MxD(+)^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ MyLD(-)^{(m)} & MxLD(-)^{(m)} & MyD(+)^{(m)} & MxD(+)^{(m)} \end{bmatrix} \quad (194)$$

$$\begin{matrix} MyL(+)^{(1)} & MxL(+)^{(1)} & VyLD^{(1)} & VxLD^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ MyL(+)^{(m)} & MxL(+)^{(m)} & VyLD^{(m)} & VxLD^{(m)} \end{matrix}\Bigg]$$

Donde cada columna representan en el orden de izquierda a derecha lo siguiente:

- Momentos en y negativo debido a carga viva más carga muerta.
- Momentos en x negativo debido a carga viva más carga muerta.
- Momentos en y positivo debido a carga muerta.
- Momentos en x positivo debido a carga muerta.
- Momentos en y positivo debido a carga viva.
- Momentos en x positivo debido a carga viva.
- Cortantes en y debido a carga viva más carga muerta.
- Cortantes en x debido a carga viva más carga muerta.

Estos valores se podrán observar en una tabla donde las filas corresponden a los números de losas y las columnas a las solicitaciones citadas anteriormente.

También se almacena el número de caso en una matriz de dimensión $mx1$ llamada nc. Que nos servirá para la obtención de los gráficos.

$$nc = \begin{bmatrix} nc^{(1)} \\ \vdots \\ nc^{(m)} \end{bmatrix} \quad (195)$$

3.8.7 Gráficos

Para finalizar se crea un bucle `for`, dentro del mismo una sentencia `if/else`. En la misma se va recorriendo la matriz nc, para ver el caso de la losa analizada, de acuerdo al caso, se van tomando los valores de la matriz mom y la matriz dim para imprimirlo sobre el gráfico representativo de la losa. En este gráfico se muestran las dimensiones de las losas y los valores de los momentos como también cortantes, con su respectivo esquema.

3.8.8 Ejemplos de prueba

Para poner a prueba el programa se utilizaron dos ejemplos del libro Diseño de estructuras de concreto de [17]. Cabe mencionar que el algoritmo trabaja con unidades de medida del sistema internacional, pero los ejemplos están en unidades inglesas, se procedió a convertir las unidades del mismo antes de utilizarlas en el programa.

El primer caso es de dos losas idénticas armadas en una dirección. Como se muestra en la figura.

Figura 35

Ejemplo de losas unidireccionales.

Ejemplo 12.1. Diseño de una losa armada en una dirección. Una losa de concreto reforzado se construye íntegramente con sus soportes y consta de dos vanos iguales, cada uno con luz libre de 15 pies. La carga viva de servicio es de 100 lb/pie^2 y se especifica concreto de 4000 lb/pulg^2 para utilizar con acero con esfuerzo de fluencia igual a 60,000 lb/pulg^2. Diseñe la losa según las disposiciones del Código ACI.

Nota. Ejemplo 12.1 Diseño de losa armada en una dirección. Año 1999. Fuente: Diseño de estructuras de concreto de [17].

Figura 36

Solución del ejemplo de losas unidireccionales.

Solución. Primero se estima el espesor de la losa con base en los espesores mínimos de la tabla 12.1: $l/28 = 15 \times 12/28 = 6.43$ pulg. Se utilizará un espesor tentativo de 6.50 pulg, para el cual el peso es $150 \times 6.50/12 = 81$ lb/pie^2. La carga viva especificada y la carga muerta calculada se multiplican por los factores de carga del Código ACI:

$$\begin{aligned} \text{Carga muerta} &= 81 \times 1.4 &&= 113 \text{ lb/pie}^2 \\ \text{Carga viva} &= 100 \times 1.7 &&= 170 \text{ lb/pie}^2 \\ \text{Total} &&&= 283 \text{ lb/pie}^2 \end{aligned}$$

En este caso, los momentos de diseño en las secciones críticas se encuentran utilizando los coeficientes de momento del ACI (ver la tabla 11.1):

$$\text{En el apoyo interior:} \quad -M = \tfrac{1}{9} \times 0.283 \times 15^2 = 7.06 \text{ klb-pie}$$

$$\text{En el centro de la luz:} \quad +M = \tfrac{1}{14} \times 0.283 \times 15^2 = 4.53 \text{ klb-pie}$$

La fuerza cortante mayorada a una distancia d de la cara del apoyo interior es

$$V_u = 1.15 \times \frac{283 \times 15}{2} - 283 \times \frac{5.50}{12} = 2310 \text{ lb}$$

Nota. Ejemplo 12.1 Diseño de losa armada en una dirección. Año 1999. Fuente: Diseño de estructuras de concreto de [17].

Introducimos los valores de entrada que son, las dimensiones de cada losa, las conectividades de una losa con otra y los niveles de cada losa.

Figura 37

Datos de partida

```
dim = np.array([[4.57, 10],  #Dimensiones en las cordenadas x, y largo y ancho en metros
                [4.57, 10]])
con = np.array([[-1, 1, -1, -1],      #Conectividad de las losas arr, der, aba, izq
                [-1, -1, -1, 0]])
ni = np.array([[0],      #Nivel de losas, el nivel de la mayoría "0" si está más abajo o más arriba "1"
                [0]])
```

Como en el libro no especifica las dimensiones más largas, ya que son losas que trabajan en la dirección más corta, se adoptaron valores que cumplan la relación de lados mayor a dos. Es decir, son valores referenciales.

El algoritmo obtiene los siguientes valores de predimensionado.

Figura 38

Espesores obtenidos del algoritmo de losas.

	Espesores predimensionados(m)
0	0.228
1	0.228

Al comparar los resultados del ejemplo, donde el espesor predimensionado de losas es de 16,33 cm, con los del programa, donde el espesor es de 22,8 cm. Se puede observar que hay una gran diferencia, esto se debe a que el programa analiza las condiciones de apoyo posterior al predimensionado y por ello toma el valor máximo (Cuando está apoyada en los dos extremos). Es decir, se obtiene un valor del lado de la seguridad pero que resulta antieconómico. Pero hay que recordar que este valor aun no corresponde al valor real del dimensionado, ya que el usuario es el que toma la decisión final en cuanto al espesor.

Seguidamente procedemos a cargar al programa los valores de las cargas permanentes y variables obtenidas en el ejemplo.

Figura 39

Carga permanente mayorada.

```
cp = np.array([5.41, 5.41]) #(kN/m2) Carga permanente mayorada
```

Figura 40

Carga variable mayorada.

```
cv = np.array([8.14, 8.14]) #(kN/m2) Carga variable mayorada
```

Con los mismos obtenemos los valores de los momentos en el apoyo intermedio y en el centro, como también el valor de la cortante.

Figura 41

Valores de Momentos y Cortantes.

	My(-) LD (kN.m/m)	Mx(-) LD (kN.m/m)	My(+) D (kN.m/m)	Mx(+) D (kN.m/m)	My(+) L (kN.m/m)	Mx(+) L (kN.m/m)	Vy LD (kN/m)	Vx LD (kN/m)
0	0.0	31.443377	0.0	10.271574	0.0	15.454826	0.0	7.79125
1	0.0	31.443377	0.0	10.271574	0.0	15.454826	0.0	7.79125

Como se puede ver, los valores del momento en el apoyo intermedio "Mx(-) LD" y la cortante "Vx LD" poseen los mismos valores que los del ejemplo. En cambio el valor del momento en el centro de la placa que corresponde a la suma de "Mx(+) D" y "Mx(+) L", Se obtiene un valor aproximadamente 27% mayor al ejemplo.

Para el segundo ejemplo, utilizamos el caso de una losa armada en dos direcciones.

Figura 42

Ejemplo de losas bidireccionales.

Ejemplo 12.2. Diseño de una losa en dos direcciones apoyada en los bordes. Un piso monolítico de concreto reforzado se compone de vanos rectangulares que miden 21 × 26 pies, como se ilustra en la figura 12.10. En todos los ejes de columnas se proporcionan vigas de 12 pulg de ancho y 24 pulg de altura; así que las luces libres para los paneles de losa en dos direcciones son 20 × 25 pies. El piso debe diseñarse para soportar una carga viva de servicio de 137 lb/pie^2 distribuida uniformemente sobre su superficie, adicional a su peso , utilizando concreto con resistencia f'_c = 3000 lb/pulg^2 y refuerzo con f_y = 60,000 lb/pulg^2. Determine el espesor requerido para la losa y el refuerzo para el panel de esquina mostrado.

Nota. Ejemplo 12.2 Diseño de losa armada en dos direcciones. Año 1999.

Fuente: Diseño de estructuras de concreto de [17].

Figura 43

Solución del ejemplo de losas bidireccionales.

Solución. El espesor mínimo de losas de este tipo se toma usualmente igual a $\frac{1}{180}$ veces el perímetro del panel:

$$h = 2(20 + 25) \times \frac{12}{180} = 6 \text{ pulg}$$

Este espesor se seleccionará como espesor tentativo. La carga muerta correspondiente es $\frac{1}{2} \times 150 = 75$ lb/pie^2. Así que, las cargas mayoradas de diseño son

Carga viva	$= 1.7 \times 137$	$=$	233 lb/pie^2
Carga muerta	$= 1.4 \times 75$	$=$	105 lb/pie^2
	Carga total	$=$	338 lb/pie^2

Con una relación de los lados del panel $m = l_a/l_b = 20/25 = 0.8$, los cálculos del momento para las franjas centrales de la losa son los siguientes:

Momentos negativos en bordes continuos (tabla 12.3)

$$M_{a,neg} = 0.071 \times 338 \times 20^2 = 9600 \text{ lb-pie} = 115{,}000 \text{ lb-pulg}$$
$$M_{b,neg} = 0.029 \times 338 \times 25^2 = 6130 \text{ lb-pie} = 73{,}400 \text{ lb-pulg}$$

Momentos positivos (tablas 12.4 y 12.5)

$$M_{a,pos,dl} = 0.039 \times 105 \times 20^2 = 1638 \text{ lb-pie} = 19{,}700 \text{ lb-pulg}$$
$$M_{a,pos,ll} = 0.048 \times 233 \times 20^2 = 4470 \text{ lb-pie} = 53{,}700 \text{ lb-pulg}$$
$$M_{a,pos,tot} = 73{,}400 \text{ lb-pulg}$$
$$M_{b,pos,dl} = 0.016 \times 105 \times 25^2 = 1050 \text{ lb-pie} = 12{,}600 \text{ lb-pulg}$$
$$M_{b,pos,ll} = 0.020 \times 233 \times 25^2 = 2910 \text{ lb-pie} = 35{,}000 \text{ lb-pulg}$$
$$M_{b,pos,tot} = 47{,}600 \text{ lb-pulg}$$

Nota. Ejemplo 12.2 Diseño de losa armada en dos direcciones. Año 1999.

Fuente: Diseño de estructuras de concreto de [17].

Introducimos los valores de entrada que son, las dimensiones de cada losa, las conectividades de una losa con otra y los niveles de cada losa. Introducimos tres losas,

pero analizamos solamente, nos interesa los resultados de la losa 0, que corresponde a la losa de la esquina que se analiza en el ejemplo.

Figura 44

Datos de partida.

```
dim = np.array([[6.1, 7.62], #Dimensiones en las cordenadas x, y largo y ancho en metros
                [6.1, 7.62],
                [6.1, 7.62]])

con = np.array([[-1, 1, 2, -1], #Conectividad de las losas arr, der, aba, izq
                [-1, -1, -1, 0],
                [0, -1, -1, -1]])

ni = np.array([[0], #Nivel de losas, el nivel de la mayoría "0" si está más abajo o más arriba "1"
               [0],
               [0]])
```

El algoritmo obtiene los siguientes valores de predimensionado.

Figura 45

Espesores obtenidos del algoritmo de losas.

	Espesores predimensionados
0	0.152
1	0.152
2	0.152

Al comparar con los resultados del ejemplo, donde el espesor predimensionado de losas es de 15,24 cm. Nos percatamos de que se obtiene el mismo valor.

Seguidamente procedemos a cargar al programa los valores de las cargas permanentes y variables obtenidas en el ejemplo.

Figura 46

Carga permanente mayorada.

```
cp = [5.03, 5.03, 5.03] #(kN/m2) Carga permanente mayorada
```

Figura 47

Carga variable mayorada.

```
cv = [11.17, 11.17, 11.17] #(kN/m2) Carga variable mayorada
```

Con los mismos obtenemos los valores de los momentos en el apoyo intermedio y en el centro, como también el valor de la cortante.

Figura 48

Valores de Momentos y Cortantes.

	My(-) LD (kN.m/m)	Mx(-) LD (kN.m/m)	My(+) D (kN.m/m)	Mx(+) D (kN.m/m)	My(+) L (kN.m/m)	Mx(+) L (kN.m/m)	Vy LD (kN/m)	Vx LD (kN/m)
0	27.328033	42.767299	4.682222	7.293591	12.992015	19.928695	17.93178	35.055163
1	0.000000	51.821986	4.387092	8.416588	12.343435	21.175603	8.66052	42.477038
2	48.032060	0.000000	63.656139	8.412658	14.937753	21.171239	31.51710	24.179776

Donde podemos comparar los valores de Momentos y Cortantes del ejemplo con lo obtenido en el algoritmo. Pudiendo observar que todos los valores son iguales.

Con todo lo visto anteriormente podemos asumir que el algoritmo es fiable y no se producirán errores que pueda poner en riesgo la seguridad de la estructura.

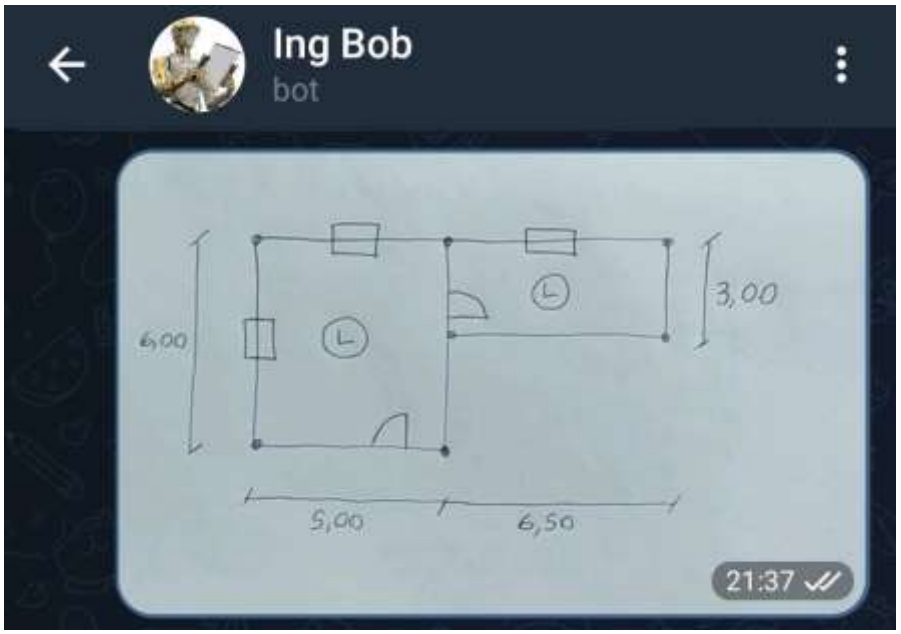
Capítulo 4

Ejemplo de Uso

Primeramente se envía al Ing. Bob a través de Telegram, la imagen que contiene el boceto del plano arquitectónico en planta.

Figura 49

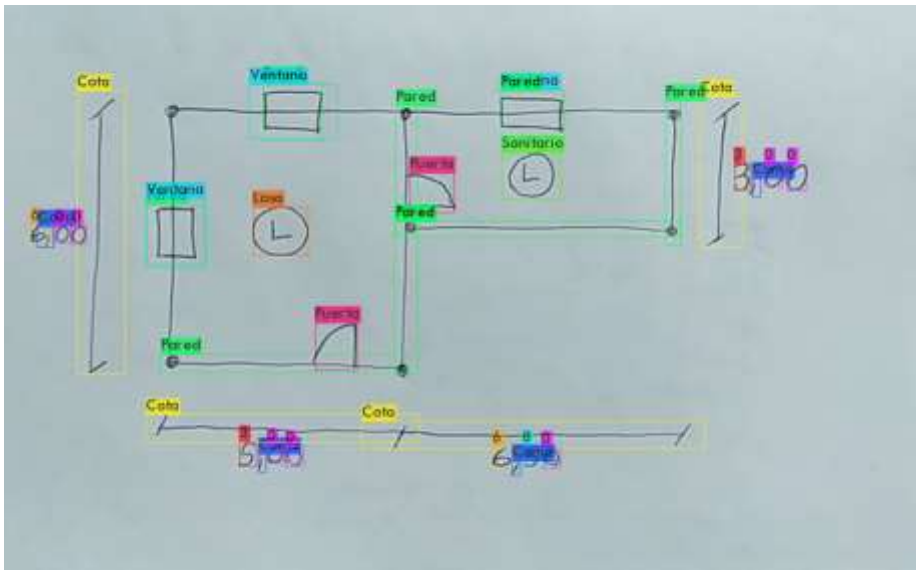
Captura de pantalla de Ing. Bob Telegram.



El sistema realiza las detecciones correspondientes y crea una carpeta en Drive nombrada con el ID del usuario que mandó la imagen, esta contiene la foto del boceto con las detecciones. Procedemos a descargar la carpeta.

Figura 50

Detecciones realizadas por Ing. Bob Telegram.



Entramos al programa de reconstrucción, En los datos de partida el usuario puede optar por las dimensiones de alturas y espesor de las mamposterías como también las dimensiones de las puertas y ventanas.

Para este ejemplo, podemos observar que al número "5" lo detectó como "3" y a un identificador de "Losa" lo detectó con un identificador de "Sanitario".

Procedemos a corregir las detecciones fallidas con LabelImg Master; cambiando las etiquetas por las que realmente corresponden a los objetos. Seguidamente procedemos a guardar los cambios en la carpeta donde se encuentra contenida la imagen y sus detecciones.

Figura 51

Captura de pantalla de edición en LabelImg Master 1.

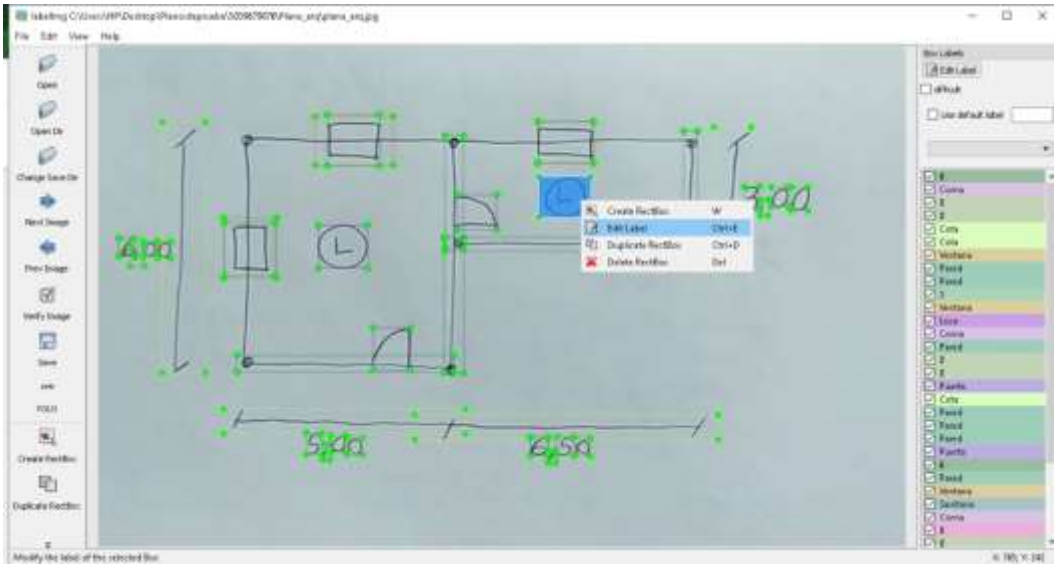


Figura 52

Captura de pantalla de edición en LabelImg Master 2.

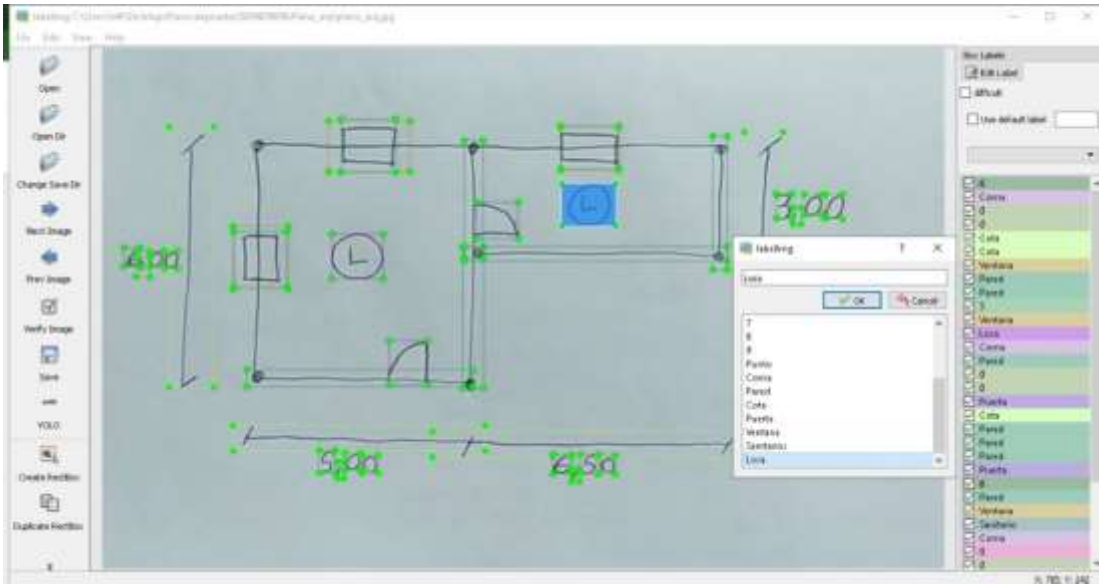
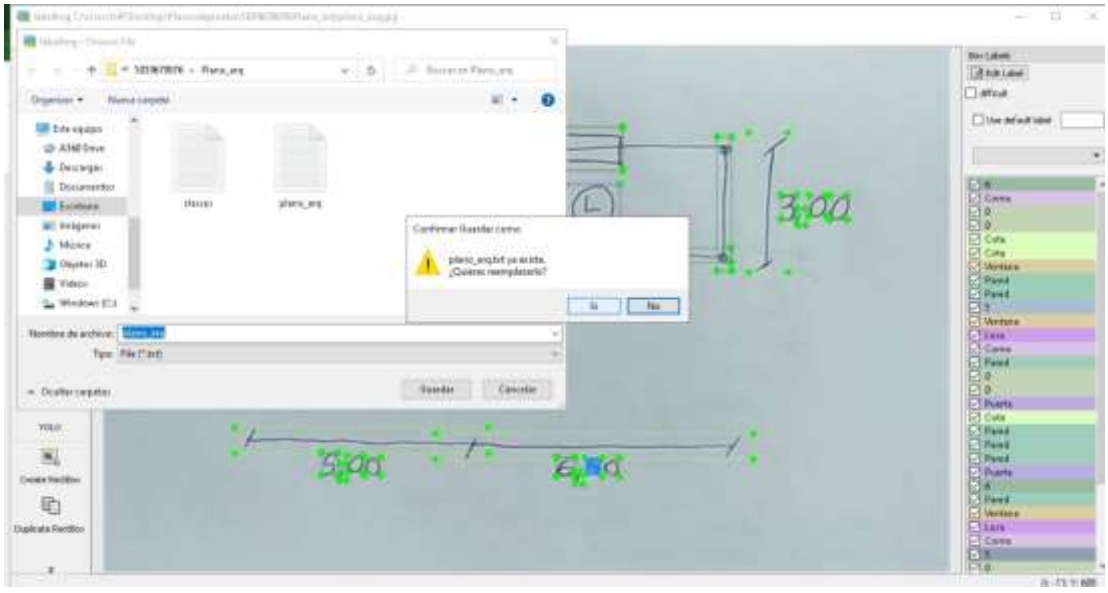


Figura 53

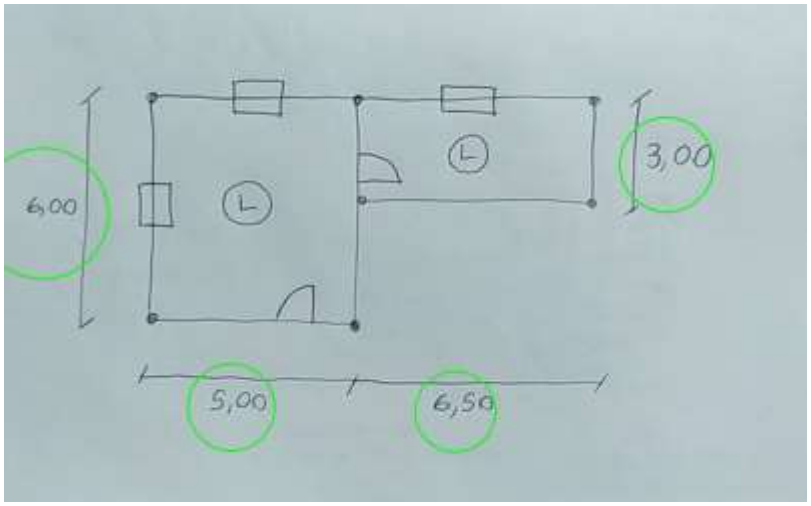
Captura de pantalla de edición en LabelImg Master 3.



Ejecutamos el algoritmo de reconstrucción, el mismo detecta los números y crea un radio que abarca las líneas de cotas donde pertenece y les asigna el valor de los números.

Figura 54

Captura de pantalla de proceso de creación de cotas.



En base a las cotas se crean los ejes donde podrían estar las mamposterías, seguidamente se asigna los puntos donde puedrían encontrarse las mamposterías.

Figura 55

Captura de pantalla de proceso de creación de mamposterías 1.

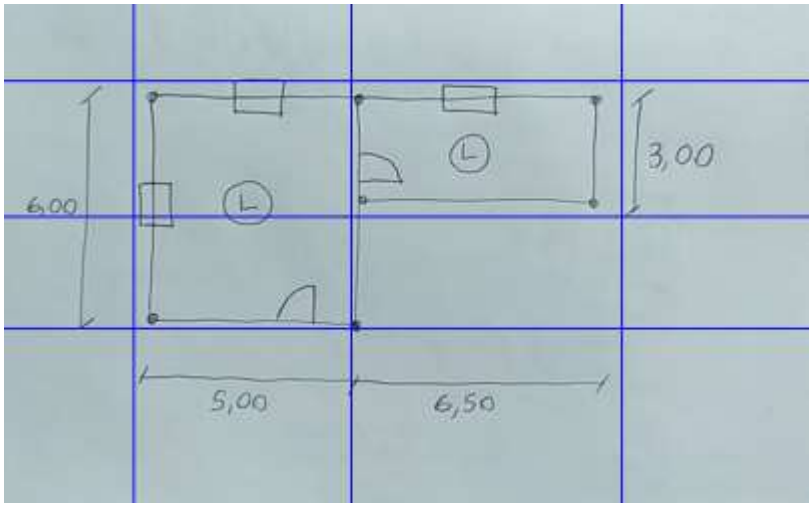
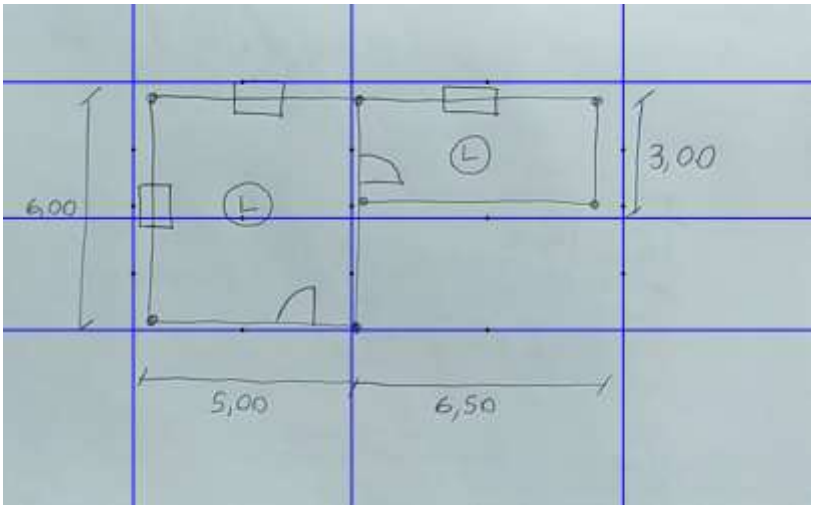


Figura 56

Captura de pantalla de proceso de creación de mamposterías 2.



A partir de estos puntos se crea un radio donde si el centro de la mampostería detectada cae en el radio, asume que hay una mampostería en el segmento del eje.

Luego elimina los segmentos de los ejes que no contienen mamposterías. Quedando solo los segmentos que delimitan las mamposterías.

Figura 57

Captura de pantalla de proceso de creación de mamposterías 3.

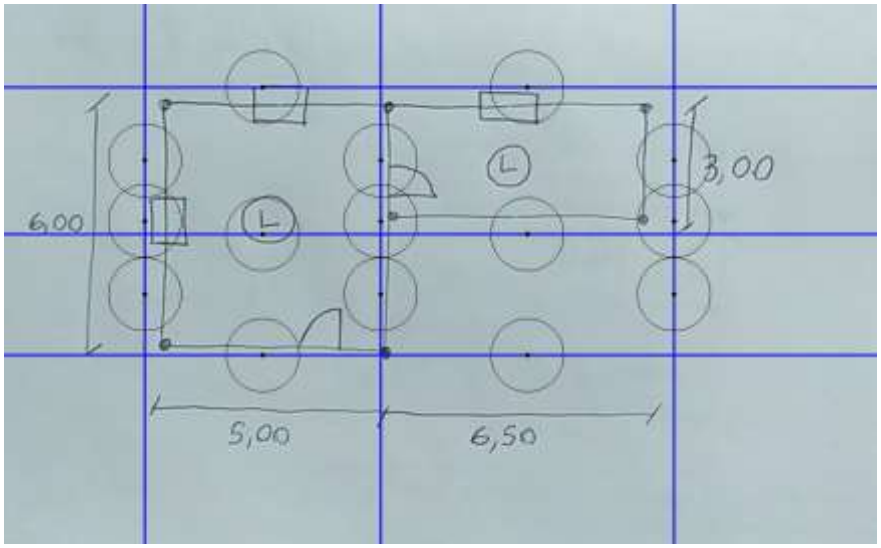
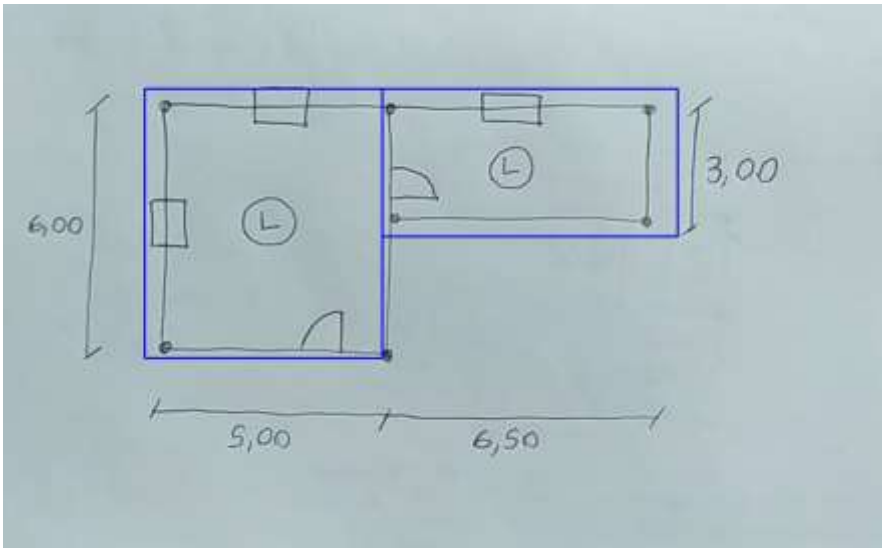


Figura 58

Captura de pantalla de proceso de creación de mamposterías 4.



Finalmente se obtienen los valores para el entorno de dibujo. Tales como las dimensiones de los ejes, las mamposterías, las puertas, ventanas y las losas. Los cuales nos servirán como datos de entrada para los algoritmos de obtención de planos CAD, obtención de cómputos como también el algoritmo de predimensionamiento y solicitaciones de losas.

Figura 59

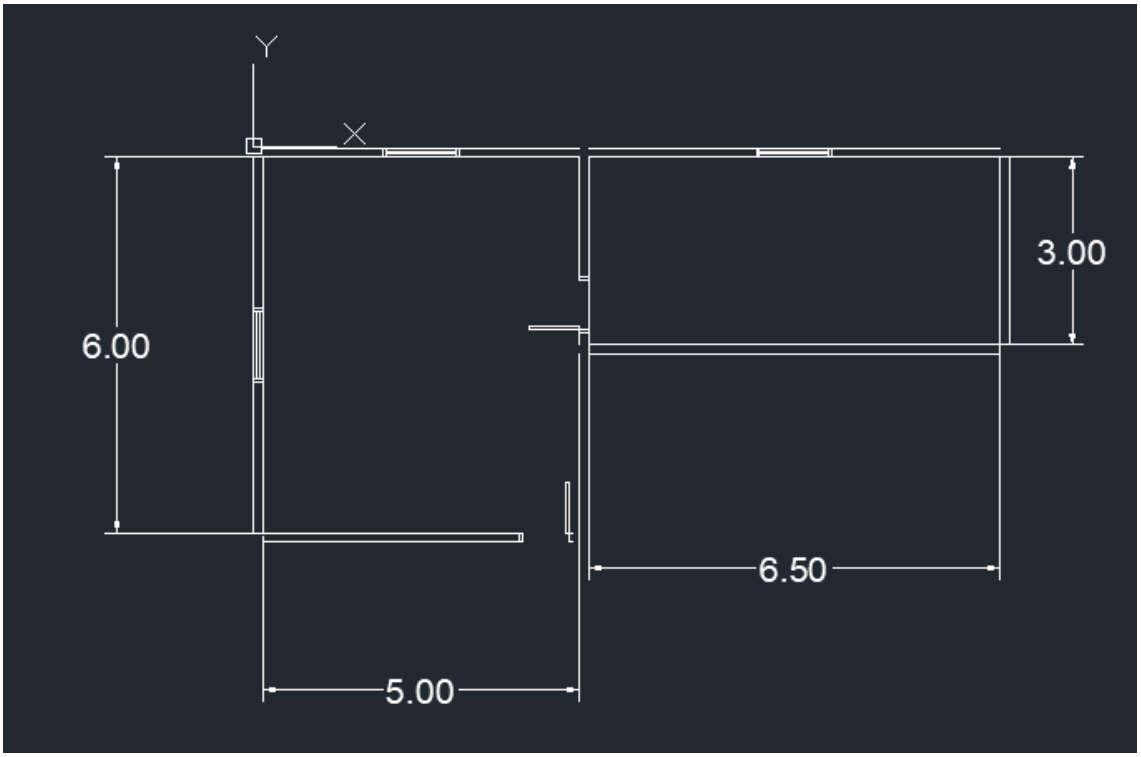
Datos de salida obtenidos en el algoritmo de Reconstrucción.

```
eh
[[123, 215, 3.15, 3.15], [123, 392, 6.15, 6.15], [338, 177, 3.0, 6.15]]

ev
[[206, 346, 5.15, 5.15], [552, 430, 6.65, 11.8], [982, 0, 0, 11.8]]

Mh
[[0.075, 0.0, 5.15, 0.15, [0, 1], [-1, 6.15]],
 [5.225, 0.0, 6.65, 0.15, [0, 1], [-1, 3.15]],
 [5.225, 3.15, 6.65, 0.15, [1, 0], [0.0, -1]],
 [0.075, 6.15, 5.15, 0.15, [1, 0], [0.0, -1]]]

Mv
[[5.15, 0.075, 3.15, 0.15, [1, 1], [0.0, 11.8]],
 [11.8, 0.075, 3.15, 0.15, [1, 0], [5.15, -1]],
 [0.0, 0.075, 6.15, 0.15, [0, 1], [-1, 5.15]],
 [5.15, 3.225, 3.0, 0.15, [1, 0], [0.0, -1]]]

Puertas
[[4.250000000000001, 6.15, 0.8, 0.15, 'h', 'iz', 'sup'],
 [5.15, 2.1500000000000004, 0.15, 0.8, 'v', 'sup', 'de']]

Ventanas
[[0.0, 2.6, 0.15, 1.1], [2.1, 0.0, 1.1, 0.15], [7.8, 0.0, 1.5, 0.15]]

Losas
[[8.55, 1.65, 6.65, 3.15, [-1, 1, -1, -1]],
 [2.6500000000000004, 3.1500000000000004, 5.15, 6.15, [-1, -1, -1, 0]]]
```

El algoritmo de obtención de planos CAD toma los valores de los ejes, mamposterías, puertas y ventanas y traza el plano en planta.

Figura 60

Plano obtenido en base al boceto inicial.



El algoritmo de obtención de cómputos toma los valores de los ejes, mamposterías, puertas y ventanas, calcula los rubros y obtiene un archivo Excel que contiene lo siguiente.

Nº	DESCRIPCION DE RUBROS	Unidad	Cantidad
I.	**TRABAJOS PRELIMINARES**		
1.1	Limpieza y Replanteo	m2	55.335
II.	**ESTRUCTURAS DE HºAº**		
2.1	Zapatas.	m3	
2.2	Pilares.	m3	0.99
2.3	Viga Inferior.	m3	1.75
2.4	Viga Superior.	m3	1.75
2.5	Losa	m3	6.31
III.	**MAMPOSTERIA**		
3.1	Mamposteria 0,15	m2	118.755
IV.	**AISLACIÓN**		

4.1	Aislación 0,15	ml	38.9
V.	**REVOQUE**		
5.1	De Paredes Interior fratachado y filtrado.	m2	125.3175
5.2	De Paredes exterior con ceresita.	m2	111.615
VI.	**CONTRAPISO**		
6.1	De cascote.	m2	
6.2	Carpeta alisada sobre contrapiso.	m2	49.5
VII.	**PISO**		
7.1	Piso Ceramico.	m2	49.5
VIII.	**ZOCALO**		

8.1	De cerámica esmaltada.	ml	39.783333

XIII.	**REVESTIMIENTOS**		
13.1	Azulejo en sshh	m2	0
XIV.	**CARPINTERIA DE MADERA**		
14.1	Marcos de madera para puertas con colocación.	un	0.00
14.2	Puerta Placa 0,70	un	0.00
XV.	**ABERTURAS DE CRISTAL**		
15.2	Puertas y ventanas de Blindex	m2	3.00
			-
XVI.	**PINTURA**		
16.1	Al látex profesional int.	m2	125.3175
16.2	Al látex profesional ext.	m2	111.615

Los valores obtenidos para cada rubro, utilizando este algoritmo, fueron comparados con resultados de otros programas de uso comercial. Donde se pudo constatar una diferencia del 1% al 2%.

Por último en el algoritmo de predimensionado y cálculo de solicitaciones, el usuario puede optar por los valores del peso específico del hormigón, peso específico de mamposterías, peso por metros cuadrados de contrapiso y piso, volumen de mamposterías en caso de tenerlas sobre las losas, peso de los techos, sobrecarga de uso. Seguidamente en base a las dimensiones y conexiones de las losas, se obtiene.

Figura 61

Espesores de losas.

	Espesores predimensionados
0	0.16
1	0.16

Figura 62

Condiciones de apoyo de losas.

	Arriba	Derecha	Abajo	Izquierda
0	Apoyada	Apoyada	Apoyada	Apoyada
1	Apoyada	Apoyada	Apoyada	Apoyada

Figura 63

Momentos y Cortantes de losas.

	My(-) LD (kN.m/m)	Mx(-) LD (kN.m/m)	My(+) D (kN.m/m)	Mx(+) D (kN.m/m)	My(+) L (kN.m/m)	Mx(+) L (kN.m/m)	Vy LD (kN/m)	Vx LD (kN/m)
0	0.0	0.0	7.252445	0.000000	2.886545	0.000000	6.463000	0.0000
1	0.0	0.0	12.929664	11.016669	5.146135	4.384744	19.880222	15.8765

Figura 64

Gráfico de Losa 0.

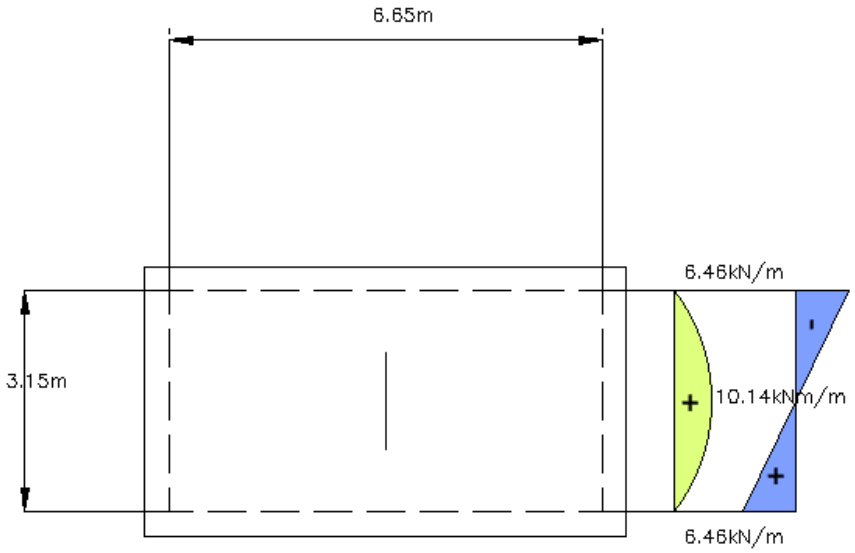
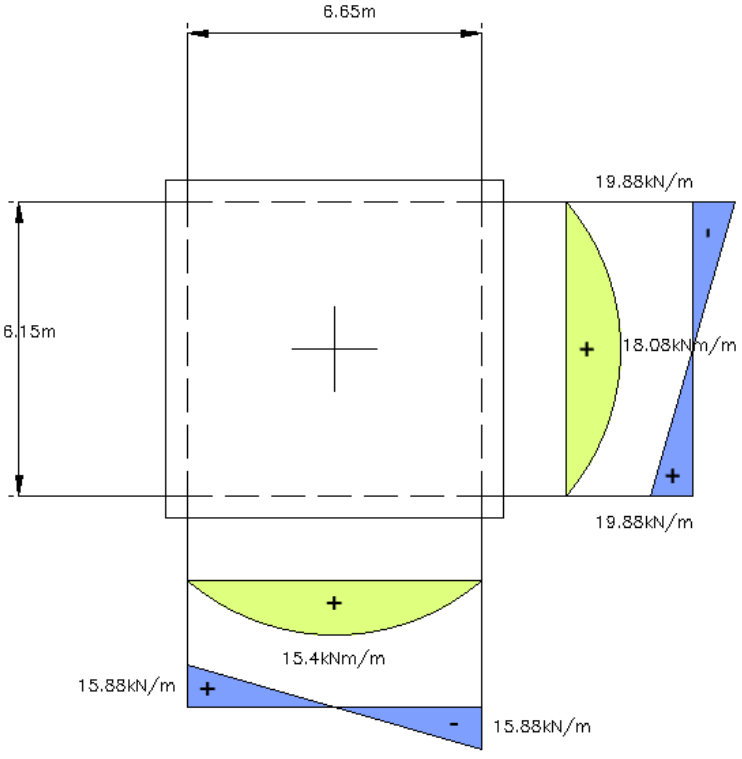


Figura 65

Gráfico de losa 1.



Capítulo 5

ANÁLISIS DE COSTOS

5.1 Análisis de costos de planos

Para este análisis se tuvo en cuenta el precio promedio de planos CAD que ofrecen diversos profesionales (Ingenieros, Arquitectos y Técnicos en construcciones civiles).

Análisis de costo de realización de planos				**UNIDAD: Metros cuadrados**
b) Mano de Obra	**Cantidad de Trabajadores**	**Horas de c/ Trabajador**	**Costo Horario Gs.**	**Costo Total Hora Horario Gs.**
Arquitecto/Ingeniero	1	8	100000	800,000
B) Total Gs.				**800,000**
C) Producción por hora=	**80**		**Costo Horario**	800,000
D) Costo Unitario de la Ejecución				10,000
E) Total Gs.				**10,000**

5.2 Análisis de costos de cómputos

Para este análisis se tuvo en cuenta el precio promedio de cómputos básicos, independientemente del método con el que se realiza, ofrecidos en el mercado diversos profesionales (Ingenieros, Arquitectos y Técnicos en construcciones civiles).

Análisis de costo realización de cómputo				**UNIDAD: Metros cuadrados**
b) Mano de Obra	**Cantidad de Trabajadores**	**Horas de c/ Trabajador**	**Costo Horario Gs.**	**Costo Total Hora Horario Gs.**
Arquitecto/Ingeniero	1	8	100000	800,000
B) Total Gs.				**800,000**
C) Producción por hora=	**67**		**Costo Horario**	800,000
D) Costo Unitario de la Ejecución				11,940
E) Total Gs.				**11,940**

5.3 Análisis de costos de cálculo estructural de losas

Para este análisis se tuvo en cuenta el precio promedio de cálculo de solicitaciones de losas y predimensionado de espesores, independientemente del método con el que se realiza, ofrecidos en el mercado diversos profesionales (Ingenieros, Arquitectos y Técnicos en construcciones civiles).

Análisis de costo cálculo estructural de losas				**UNIDAD: Metros cuadrados**
b) Mano de Obra	**Cantidad de Trabajadores**	**Horas de c/ Trabajador**	**Costo Horario Gs.**	**Costo Total Hora Horario Gs.**
Ingeniero	1	8	200000	1,600,000
B) Total Gs.				**1,600,000**
C) Producción por hora=		**100**	**Costo Horario**	1,600,000
D) Costo Unitario de la Ejecución				16,000
E) Total Gs.				**16,000**

En base al análisis de costos. Suponiendo la realización de un plano de 100m2. Se deduce lo siguiente.

- Con la realización del plano en planta CAD, se estaría obteniendo un ingreso de 1.000.000 Gs en aproximadamente 1,25 horas.
- Con la realización del cómputo, estaría obteniendo un ingreso de 1.200.000 Gs en aproximadamente 1,5 horas.
- Con la realización del cálculo de solicitaciones y predimensionado estaría obteniendo un ingreso de 1.600.000 Gs en aproximadamente 1 hora.

Tomando los puntos anteriores, se observa que el profesional estaría obteniendo un ingreso de 3.800.000 Gs en 3,75 horas.

En cambio con la utilización del programa que se desarrolló en este proyecto. Se logra optimizar la realización de lo mencionado anteriormente, pudiendo hacerse 100m2 y obtener un ingreso de 3.800.000 Gs en 1 hora.

Capítulo 6

CONCLUSIONES

Se entrenó con éxito al sistema, logrando en promedio un 87% de aciertos en las detecciones de los objetos componentes en los bocetos arquitectónicos.

Se desarrolló un algoritmo de reconstrucción, se observó que el mismo es capaz de obtener valores en base a las detecciones y asignarlos a cada uno de los elementos del plano.

Para pasar el boceto a un dibujo digitalizado, se desarrolló un algoritmo que interpreta algunos de los valores del algoritmo de reconstrucción y traza los planos en planta CAD. Se pudo ver que este plano posee las características que aparecen en el boceto y que anteriormente fueron definidas en la preparación de base de datos y el entrenamiento del detector.

Para el cálculo de cómputos, se desarrolló un algoritmo que toma los valores del plano de reconstrucción y hace un cómputo básico de los distintos elementos constructivos. Se realizaron ejemplos, se calcularon las cantidades a mano y se

compararon con los resultados del algoritmo, estos resultados coincidían o tenían una variación de aproximadamente 10% más que las cantidades calculadas a mano.

Se desarrolló un algoritmo que toma los valores de las losas obtenidos en el algoritmo de reconstrucción y predimensiona los espesores como también calcula las solicitaciones (Momentos y Cortantes). Con los ejemplos, vimos que los valores obtenidos por este algoritmo coinciden o están un poco por encima. Es decir, por el lado de la seguridad.

Como pudimos observar en el análisis de costos. La aplicación de este método, resulta un ahorro de tiempo y por ende la obtención de un mayor beneficio económico para el profesional. Como contraparte posee la desventaja de que los resultados que obtiene no son óptimos, pero pueden ser mejorados con el desarrollo de algoritmos de ajuste.

6.1 Recomendaciones y trabajos futuros

El programa realizado para este proyecto presenta una precisión de casi el 90 por ciento en la detección de objetos. Es una precisión bastante alta y por ende aceptable, pero aún se puede aumentar el porcentaje con mayor variedad de datos y más horas de entrenamiento.

A pesar de ser un sistema, intuitivo de usar para la mayoría de las personas, se recomienda su uso solo a profesionales del área de construcciones civiles. Ya que los mismos están capacitados para la interpretación de los resultados obtenidos.

Se realizaron pruebas para la aplicación del programa en distintos softwares como Sketchup, Cype y Revit donde se vieron resultados alentadores

APÉNDICES

A.1 Cargas permanentes

Elemento	Peso unitario kN/m^2 (1)	Peso unitario kN/m^3 (1)
• **Cubiertas**		
Chapa ondulada de fibra orgánica, sin estructura sostén	0,03	
Chapa acanalada de sección ondulada o trapezoidal de aluminio sin estructura de sostén		
0,6 mm de espesor (onda chica)	0,025	
0,8 mm de espesor (onda grande)	0,03	
1,0 mm de espesor (onda grande)	0,04	
Chapa acanalada de perfil ondulado o trapezoidal de acero zincado o aluminizado		
0,4 mm de espesor	0,04	
0,7 mm de espesor	0,07	
1,0 mm de espesor	0,10	
Chapa de cobre de 0,6 mm de espesor, sobre entablonado, incluido éste	0,25	
Chapa de zinc de 0,7mm de espesor, sobre entablonado, incluido éste	0,25 (*)	
Chapa de plástico reforzado, espesor 1,5 mm sobre enlistonado incluido éste	0,15	
Cubierta impermeabilizante con base de tela o cartón asfáltico de siete capas	0,10	
Doble chapa de aluminio con núcleo de poliestireno expandido	0,13	
Teja asfáltica sobre enlistonado, incluido éste	0,20	
Teja cerámica tipo español, colonial o árabe, sobre entablonado incluido éste	0,9 (*)	
Teja cerámica tipo de Marsella o francés, sobre entablonado, incluido éste	0,65 (*)	
Teja cerámica tipo flamenco, sobre entablonado, incluido éste	0,7 (*)	
Teja cerámica tipo normando, sobre entablonado, incluido éste	0,8 (*)	
• **Pisos y Contrapisos**		
Adoquín de madera 76 mm, sobre mastic, sin relleno	0,48	
Adoquín de madera 76 mm, sobre base de mortero de 13mm	0,77	
Baldosa cerámica, 12 mm de espesor	0,28	23
Baldosa de gres cerámico, 20 mm de espesor	0,38	
Baldosa de vidrio plana sobre estructura de acero	0,45	
Baldosa vinílica, 3,2 mm de espesor	0,07	
Baldosa de mortero de cemento		22
Baldosón granítico, 38 mm de espesor	0,90	

Tabla de Pesos unitarios de los materiales y conjuntos funcionales de construcción [5].

Elemento	Peso unitario kN/m^2 (1)	Peso unitario kN/m^3 (1)
• **Hormigones**		
Hormigón de cemento pórtland, arena y canto rodado o piedra partida		
sin armar		23,5
armado		25
Hormigón de cemento pórtland, arena y agregado basáltico		25
Hormigón de cemento pórtland, arena y cascote		18
Hormigón de cemento pórtland, arena y mineral de hierro		36
Hormigón de cemento pórtland, arena y arcilla expandida		8 a 20
Hormigón de cal, arena y cascote		16
Hormigón con agregado de poliestireno de alta densidad		5 a 12
• **Mampostería**		
Con revoque o completa, mortero a la cal o cemento		
Bloque hueco de hormigón		17
Bloque hueco de hormigón liviano		15
Ladrillo cerámico macizo común		17
Ladrillo hueco cerámico portante, % huecos menor que 60		12
Ladrillo hueco cerámico no portante,% huecos mayor que 60		10,5
Ladrillo refractario		26
Ladrillo de yeso		12
Piedra arenisca		26
Piedra granítica		26
Sin revoque, mortero a la cal o cemento		
Bloque hueco de hormigón		15
Bloque hueco de hormigón liviano		12,5
Ladrillo cerámico macizo común		16
Ladrillo hueco cerámico portante, % huecos menor que 60		10
Ladrillo hueco cerámico no portante, % huecos mayor que 60		8

Tabla de Pesos unitarios de los materiales y conjuntos funcionales de construcción [5].

A.2 Cargas variables

Destino	Uniforme (kN/m^2)	Concentrada (kN)
Bowling, billar y áreas recreacionales similares	4	
Cielorrasos con posibilidad de almacenamiento áreas de almacenamiento liviano áreas de almacenamiento ocasional accesibles con fines de mantenimiento	(1) 1 0,5	 1
Cocinas viviendas otros destinos	(5) 2 4	
Comedores, restaurantes y confiterías	5	
Comercio (Negocios) venta al menudeo planta baja pisos superiores comercio al por mayor, todos los pisos	 5 4 6	 4,5 4,5 4,5
Escuelas aulas corredores en pisos superiores a planta baja corredores en planta baja	 3 4 5	 4,5 4,5 4,5
Estrados y tribunas Estadios sin asientos fijos con asientos fijos (ajustados al piso)	5 (artíc. 4.6.2.) artículo 4.6.2. 5 3	
Fábricas manufactura liviana manufactura pesada	artículo 4.13. 6 12	 9 14
Gimnasios, áreas principales y balcones	5 (3)	
Hospitales salas de operaciones, laboratorios habitaciones privadas salas corredores en pisos superiores a planta baja	 3 2 2 4	 4,5 4,5 4,5 4,5
Oficinas, (Edificios para Oficinas) salas de computación y archivo se diseñarán para cargas mayores basadas en el destino previsto salones de entrada y corredores de planta baja oficinas corredores en pisos superiores a planta baja	 5 2,5 4	 9 9 9

Sobrecargas mínimas uniformemente distribuidas y sobrecargas mínimas concentradas [5].

A.3 Método ACI

Momento positivo Vanos extremos El extremo discontinuo no está restringido ..... $w_u \ell_n^2/11$ El extremo discontinuo es monolítico con el apoyo ..... $w_u \ell_n^2/14$ Vanos interiores ..... $w_u \ell_n^2/16$
Momento negativo en la cara exterior del primer apoyo interior Dos vanos ..... $w_u \ell_n^2/9$ Más de dos vanos ..... $w_u \ell_n^2/10$
Momento negativo en las demás caras de apoyos interiores ..... $w_u \ell_n^2/11$
Momento negativo en la cara de todos los apoyos para: Losas con luces que no excedan de 3 m, y vigas en las cuales la relación entre la suma de las rigideces de las columnas y la rigidez de la viga exceda de 8 en cada extremo del vano ..... $w_u \ell_n^2/12$
Momento negativo en la cara interior de los apoyos exteriores para los elementos construidos monolíticamente con sus apoyos Cuando el apoyo es una viga de borde ..... $w_u \ell_n^2/24$ Cuando el apoyo es una columna ..... $w_u \ell_n^2/16$
Cortante en elementos extremos en la cara del primer apoyo interior ..... $1.15 w_u \ell_n/2$
Cortante en la cara de todos los demás apoyos ..... $w_u \ell_n/2$

Método ACI para cálculo de momentos y cortantes en vigas y losas unidireccionales [3].

A.4 Tablas de Marcus

$M_{a,neg} = C_{a,neg} w l_a^2$

$M_{b,neg} = C_{a,neg} w l_b^2$

donde w = carga muerta más viva uniforme total

Relación $m = \frac{l_a}{l_b}$		Caso 1	Caso 2	Caso 3	Caso 4	Caso 5	Caso 6	Caso 7	Caso 8	Caso 9
1.00	$C_{a,neg}$		0.045		0.050	0.075	0.071		0.033	0.061
	$C_{b,neg}$		0.045	0.076	0.050			0.071	0.061	0.033
0.95	$C_{a,neg}$		0.050		0.055	0.079	0.075		0.038	0.065
	$C_{b,neg}$		0.041	0.072	0.045			0.067	0.056	0.029
0.90	$C_{a,neg}$		0.055		0.060	0.080	0.079		0.043	0.068
	$C_{b,neg}$		0.037	0.070	0.040			0.062	0.052	0.025
0.85	$C_{a,neg}$		0.060		0.066	0.082	0.083		0.049	0.072
	$C_{b,neg}$		0.031	0.065	0.034			0.057	0.046	0.021
0.80	$C_{a,neg}$		0.065		0.071	0.083	0.086		0.055	0.075
	$C_{b,neg}$		0.027	0.061	0.029			0.051	0.041	0.017
0.75	$C_{a,neg}$		0.069		0.076	0.085	0.088		0.061	0.078
	$C_{b,neg}$		0.022	0.056	0.024			0.044	0.036	0.014
0.70	$C_{a,neg}$		0.074		0.081	0.086	0.091		0.068	0.081
	$C_{b,neg}$		0.017	0.050	0.019			0.038	0.029	0.011
0.65	$C_{a,neg}$		0.077		0.085	0.087	0.093		0.074	0.083
	$C_{b,neg}$		0.014	0.043	0.015			0.031	0.024	0.008
0.60	$C_{a,neg}$		0.081		0.089	0.088	0.095		0.080	0.085
	$C_{b,neg}$		0.010	0.035	0.011			0.024	0.018	0.006
0.55	$C_{a,neg}$		0.084		0.092	0.089	0.096		0.085	0.086
	$C_{b,neg}$		0.007	0.028	0.008			0.019	0.014	0.005
0.50	$C_{a,neg}$		0.086		0.094	0.090	0.097		0.089	0.088
	$C_{b,neg}$		0.006	0.022	0.006			0.014	0.010	0.003

[a] Un borde achurado indica que la losa continúa a través o se encuentra empotrada en el apoyo; un borde sin marcas indica un apoyo donde la resistencia torsional es despreciable.

Coeficientes de Momentos negativos debidos a carga total en losas [17].

$M_{a,pos,dl} = C_{a,dl} w l_a^2$

$M_{b,pos,dl} = C_{b,dl} w l_b^2$

donde w = carga muerta uniforme total

Relación $m = \frac{l_a}{l_b}$		Caso 1	Caso 2	Caso 3	Caso 4	Caso 5	Caso 6	Caso 7	Caso 8	Caso 9
1.00	$C_{a,dl}$	0.036	0.018	0.018	0.027	0.027	0.033	0.027	0.020	0.023
	$C_{b,dl}$	0.036	0.018	0.027	0.027	0.018	0.027	0.033	0.023	0.020
0.95	$C_{a,dl}$	0.040	0.020	0.021	0.030	0.028	0.036	0.031	0.022	0.024
	$C_{b,dl}$	0.033	0.016	0.025	0.024	0.015	0.024	0.031	0.021	0.017
0.90	$C_{a,dl}$	0.045	0.022	0.025	0.033	0.029	0.039	0.035	0.025	0.026
	$C_{b,dl}$	0.029	0.014	0.024	0.022	0.013	0.021	0.028	0.019	0.015
0.85	$C_{a,dl}$	0.050	0.024	0.029	0.036	0.031	0.042	0.040	0.029	0.028
	$C_{b,dl}$	0.026	0.012	0.022	0.019	0.011	0.017	0.025	0.017	0.013
0.80	$C_{a,dl}$	0.056	0.026	0.034	0.039	0.032	0.045	0.045	0.032	0.029
	$C_{b,dl}$	0.023	0.011	0.020	0.016	0.009	0.015	0.022	0.015	0.010
0.75	$C_{a,dl}$	0.061	0.028	0.040	0.043	0.033	0.048	0.051	0.036	0.031
	$C_{b,dl}$	0.019	0.009	0.018	0.013	0.007	0.012	0.020	0.013	0.007
0.70	$C_{a,dl}$	0.068	0.030	0.046	0.046	0.035	0.051	0.058	0.040	0.033
	$C_{b,dl}$	0.016	0.007	0.016	0.011	0.005	0.009	0.017	0.011	0.006
0.65	$C_{a,dl}$	0.074	0.032	0.054	0.050	0.036	0.054	0.065	0.044	0.034
	$C_{b,dl}$	0.013	0.006	0.014	0.009	0.004	0.007	0.014	0.009	0.005
0.60	$C_{a,dl}$	0.081	0.034	0.062	0.053	0.037	0.056	0.073	0.048	0.036
	$C_{b,dl}$	0.010	0.004	0.011	0.007	0.003	0.006	0.012	0.007	0.004
0.55	$C_{a,dl}$	0.088	0.035	0.071	0.056	0.038	0.058	0.081	0.052	0.037
	$C_{b,dl}$	0.008	0.003	0.009	0.005	0.002	0.004	0.009	0.005	0.003
0.50	$C_{a,dl}$	0.095	0.037	0.080	0.059	0.039	0.061	0.089	0.056	0.038
	$C_{b,dl}$	0.006	0.002	0.007	0.004	0.001	0.003	0.007	0.004	0.002

[a] Un borde achurado indica que la losa continúa a través o se encuentra empotrada en el apoyo; un borde sin marcas indica un apoyo donde la resistencia torsional es despreciable.

Coeficientes de Momentos positivos debidos a carga muerta en losas [17].

$M_{a,pos,II} = C_{a,II} w l_a^2$

$M_{b,pos,II} = C_{b,II} w l_b^2$

donde w = carga viva uniforme total

Relación $m = \frac{l_a}{l_b}$		Caso 1	Caso 2	Caso 3	Caso 4	Caso 5	Caso 6	Caso 7	Caso 8	Caso 9
1.00	$C_{a,II}$	0.036	0.027	0.027	0.032	0.032	0.035	0.032	0.028	0.030
	$C_{b,II}$	0.036	0.027	0.032	0.032	0.027	0.032	0.035	0.030	0.028
0.95	$C_{a,II}$	0.040	0.030	0.031	0.035	0.034	0.038	0.036	0.031	0.032
	$C_{b,II}$	0.033	0.025	0.029	0.029	0.024	0.029	0.032	0.027	0.025
0.90	$C_{a,II}$	0.045	0.034	0.035	0.039	0.037	0.042	0.040	0.035	0.036
	$C_{b,II}$	0.029	0.022	0.027	0.026	0.021	0.025	0.029	0.024	0.022
0.85	$C_{a,II}$	0.050	0.037	0.040	0.043	0.041	0.046	0.045	0.040	0.039
	$C_{b,II}$	0.026	0.019	0.024	0.023	0.019	0.022	0.026	0.022	0.020
0.80	$C_{a,II}$	0.056	0.041	0.045	0.048	0.044	0.051	0.051	0.044	0.042
	$C_{b,II}$	0.023	0.017	0.022	0.020	0.016	0.019	0.023	0.019	0.017
0.75	$C_{a,II}$	0.061	0.045	0.051	0.052	0.047	0.055	0.056	0.049	0.046
	$C_{b,II}$	0.019	0.014	0.019	0.016	0.013	0.016	0.020	0.016	0.013
0.70	$C_{a,II}$	0.068	0.049	0.057	0.057	0.051	0.060	0.063	0.054	0.050
	$C_{b,II}$	0.016	0.012	0.016	0.014	0.011	0.013	0.017	0.014	0.011
0.65	$C_{a,II}$	0.074	0.053	0.064	0.062	0.055	0.064	0.070	0.059	0.054
	$C_{b,II}$	0.013	0.010	0.014	0.011	0.009	0.010	0.014	0.011	0.009
0.60	$C_{a,II}$	0.081	0.058	0.071	0.067	0.059	0.068	0.077	0.065	0.059
	$C_{b,II}$	0.010	0.007	0.011	0.009	0.007	0.008	0.011	0.009	0.007
0.55	$C_{a,II}$	0.088	0.062	0.080	0.072	0.063	0.073	0.085	0.070	0.063
	$C_{b,II}$	0.008	0.006	0.009	0.007	0.005	0.006	0.009	0.007	0.006
0.50	$C_{a,II}$	0.095	0.066	0.088	0.077	0.067	0.078	0.092	0.076	0.067
	$C_{b,II}$	0.006	0.004	0.007	0.005	0.004	0.005	0.007	0.005	0.004

[a] Un borde achurado indica que la losa continúa a través o se encuentra empotrada en el apoyo; un borde sin marcas indica un apoyo donde la resistencia torsional es despreciable.

Coeficientes de Momentos positivos debidos a carga viva en losas [17].

Relación de la carga W que se transmite en las direcciones l_a y l_b para calcular el cortante en la losa y las cargas en los apoyos[a]

Relación $m=\frac{l_a}{l_b}$		Caso 1	Caso 2	Caso 3	Caso 4	Caso 5	Caso 6	Caso 7	Caso 8	Caso 9
1.00	W_a	0.50	0.50	0.17	0.50	0.83	0.71	0.29	0.33	0.67
	W_b	0.50	0.50	0.83	0.50	0.17	0.29	0.71	0.67	0.33
0.95	W_a	0.55	0.55	0.20	0.55	0.86	0.75	0.33	0.38	0.71
	W_b	0.45	0.45	0.80	0.45	0.14	0.25	0.67	0.62	0.29
0.90	W_a	0.60	0.60	0.23	0.60	0.88	0.79	0.38	0.43	0.75
	W_b	0.40	0.40	0.77	0.40	0.12	0.21	0.62	0.57	0.25
0.85	W_a	0.66	0.66	0.28	0.66	0.90	0.83	0.43	0.49	0.79
	W_b	0.34	0.34	0.72	0.34	0.10	0.17	0.57	0.51	0.21
0.80	W_a	0.71	0.71	0.33	0.71	0.92	0.86	0.49	0.55	0.83
	W_b	0.29	0.29	0.67	0.29	0.08	0.14	0.51	0.45	0.17
0.75	W_a	0.76	0.76	0.39	0.76	0.94	0.88	0.56	0.61	0.86
	W_b	0.24	0.24	0.61	0.24	0.06	0.12	0.44	0.39	0.14
0.70	W_a	0.81	0.81	0.45	0.81	0.95	0.91	0.62	0.68	0.89
	W_b	0.19	0.19	0.55	0.19	0.05	0.09	0.38	0.32	0.11
0.65	W_a	0.85	0.85	0.53	0.85	0.96	0.93	0.69	0.74	0.92
	W_b	0.15	0.15	0.47	0.15	0.04	0.07	0.31	0.26	0.08
0.60	W_a	0.89	0.89	0.61	0.89	0.97	0.95	0.76	0.80	0.94
	W_b	0.11	0.11	0.39	0.11	0.03	0.05	0.24	0.20	0.06
0.55	W_a	0.92	0.92	0.69	0.92	0.98	0.96	0.81	0.85	0.95
	W_b	0.08	0.08	0.31	0.08	0.02	0.04	0.19	0.15	0.05
0.50	W_a	0.94	0.94	0.76	0.94	0.99	0.97	0.86	0.89	0.97
	W_b	0.06	0.06	0.24	0.06	0.01	0.03	0.14	0.11	0.03

[a] Un borde achurado indica que la losa continúa a través o se encuentra empotrada en el apoyo; un borde sin marcas indica un apoyo donde la resistencia torsional es despreciable.

Coeficientes de distribución de Cortantes debidas a carga total en losas [17].

Referencias Bibliográficas

[1] L. Rouhianen, Inteligencia Artificial 101 Cosas que debes saber hoy sobre el Futuro, 2018.

[2] D. Alemán, «Técnicas de inteligencia artificial aplicadas a problemas de ingeniería civil,» *Revista de Arquitectura e Ingeniería,* 2017.

[3] N. Fatemi, J. Nikolic y F. Moreno, «Modelling a new workflow based on emotional analysis of floor-plans using machine learning algorithams and semiotics,» 2019.

[4] M. Massiris, C. Delrieux y J. Á. Fernández, «Detección de equipos de protección personal mediante red neuronal convolucional YOLO,» 2018.

[5] R. Muñoz Manso, *Sistema de visión artificial para la detección y lectura de matrículas,* 2014.

[6] J. Berger, A. Tamada y R. Feltan, «Aplicación móvil para la identificación de variedades de Manihot esculenta Crantz cultivadas en Misiones mediante técnicas de deep learning,» 2019.

[7] L. L. Zambrano Salazar, *Elaboración de un software en base a Matlab para el cálculo y diseño de losas macizas y nervadas,* 2017.

[8] J. Redmond, S. Divvala, R. Girshick y A. Farhadi, «You only look once: Unified, real-time object detection,» de *Computer vision and pattern recognition (pp. 779–788).*, 2016.

[9] ACI 318, *Requisitos de Reglamento para Concreto Estructural (ACI 318S-05) y Comentario (ACI 318SR-05),* 2005.

[10] ACI 318, *Código de construcción para hormigón reforzado,* 1963.

[11] CIRSOC 101, *Reglamento Argentino de cargas permanentes y sobrecargas mínimas de Diseño para Edificios y otras Estructuras,* 2005.

[12] S. Haykin, Neural Networks, 1999.

[13] P. Isasi Viñuela y I. M. Galván León, Redes de Neuronas Artificiales Un Enfoque Práctico, 2004.

[14] D. Calvo, «diegocalvo.es,» 2018. [En línea].

[15] T. E. Oliphant, *numpy,* 2006.

[16] W. McKinney, *pandas,* 2009.

[17] Intel Corporation, *Open CV,* 2000.

[18] E. Gazoni y C. Clark, *openpyxl,* 2013.

[19] R. Haritonov, *pytoautocad,* 2012.

[20] MACTI, *Uso de materiales didacticos mediante Google Colab.*

[21] J. Redmon, D. Santosh, R. Girshick y A. Farhadi, *You Only Look Once: Unified, Real-Time Object Detection,* 2016.

[22] A. Kathuria, «towardsdatascience.com,» 2018. [En línea].

[23] J. Redmon y A. Farhadi, *YOLOv3: An Incremental Improvement,* 2018.

[24] V. Rajput, «Medium.com,» 2021. [En línea].

[25] A. H. Nilson, Diseño de Estructuras de Concreto, 1999.

[26] J. L. Macchia, Cómputos, Costos y Presupuestos, 2009.

[27] Q. Nguyen, «towardsdatascience.com,» 2020. [En línea].