Algoritmo de detección y lectura de dígitos de siete segmentos

Esteban J. Odetti, Andrés F. Pozzer
estebanode@gmail.com, andres.pozzer.ap@gmail.com
Procesamiento digital de imágenes – Departamento de Informática – FICH UNL

***Resumen*— En la vida cotidiana nos encontramos frente a una gran cantidad de dispositivos digitales con pantallas de siete segmentos, pero no todos poseen la capacidad de transferir o exportar la información de diversas maneras para que personas con discapacidades visuales puedan acceder a la misma; es por eso que se implementó un algoritmo semiautomático para lectura de estos dígitos, a través de una imagen. Los resultados fueron satisfactorios logrando una precisión del 83% en promedio bajo condiciones de luz deseables. Proyectos anteriores a este dieron resultados con alta precisión, pero son solo para un tipo de dispositivo, por lo que se centró en otorgar generalidad de detección.**

***Palabras claves*— pantallas, dígitos, lectura.**

I. INTRODUCCIÓN

Las pantallas de siete segmentos son una de las maneras más simples de dar información numérica. Las mismas están presentes en muchos de los dispositivos tecnológicos que habitualmente se utilizan en la vida diaria, ya sea en un reloj, microondas, medidores de presión arterial, balanzas, etc. por lo que son de uso cotidiano. Según el INDEC, en la Argentina, existe un 25% de personas con discapacidad visual, entre ellas un 96,4% posee mucha dificultad para ver, mientras que el restante 3,6% posee dificultad total[1]. Habitualmente, este grupo de personas utiliza la tecnología y dada su condición, poseen problemas al utilizar algunos dispositivos. En este sentido, una herramienta que permita a estas personas obtener la información de otra manera sería muy útil. En este contexto, con el fin de facilitar la información esclareciendo los resultados obtenidos en las pantallas, se desarrolló una aplicación que facilite mediante la captura de una imagen a través de una cámara, aplicando técnicas de procesamiento digital de imágenes, la interpretación de los dispositivos, dictando por voz el resultado que se muestra en pantalla. Este proyecto es libre y gratuito y se definió de forma conjunta con la Asociación Santafesina Nueva Cultura[2] a través de diferentes reuniones. Proyectos anteriores a este dieron resultados con alta precisión, pero son solo para un tipo de dispositivo, por lo que se centró en otorgar generalidad de detección. [1-3]

II. MATERIALES Y METODOLOGÍAS

En esta sección se describe el conjunto de datos y las condiciones para obtener las imágenes. Las herramientas utilizadas fueron Python v3.9 con las bibliotecas Numpy, Sciki-Image, imultis[3], opencv y pyttsx3[4]. [4-7]

A. CONJUNTO DE DATOS

Está conformado por una serie de imágenes y videos capturados por los autores de este proyecto. Ejemplo de estos dispositivos son medidores de presión arterial y una balanza de uso doméstico. Las imágenes de la balanza de uso comercial retroiluminada fueron provistas por personas pertenecientes a la Asociación Santafesina Nueva Cultura. Las mismas fueron obtenidas con cámaras de distintos smartphones. Para la balanza retroiluminada se contaron con 6 fotos y para el resto de los equipos conjunto de datos es de 158 fotos cada uno.

Figura 1: Ejemplos de imágenes de dispositivos.

[1] https://www.indec.gob.ar/ftp/cuadros/poblacion/estudio_discapacidad_12_18.pdf
[2] https://nuevaculturasf.org.ar/#!/-inicio/
[3] https://pypi.org/project/imutils/
[4] https://pypi.org/project/pyttsx3/

B. Condiciones lumínicas.

Las condiciones a las que fueron expuestos los dispositivos al obtener las imágenes se presentan en la tabla 1:

Tabla 1: Tabla de condiciones lumínicas.

Dispositivo	**Tipo de luz**	**Flash**	**Fuente de luz**	**Pantalla retro iluminada**	**Distancia al dispositivo**
Tensiómetro clásico	Foco led	No	Única, artificial, indirecta	No	30 cm
Tensiómetro muñeca	Foco led	No	Única, artificial, indirecta	No	10 cm
Balanza hogareña	Foco halógeno	No	Única, artificial, directa	No	15 cm
Balanza comercial	Luz ambiente	No	Única, natural, indirecta	Si	30 cm

III. ALGORITMO PROPUESTO

Inicialmente, se desarrollaron métodos que se ajustaban para todas las imágenes de entrada, pero después de realizar pruebas se verificó que su generalización no era realizable. Por lo tanto, se decidió generar plantillas simples con configuraciones específicas para cada dispositivo. De esta forma, al cargar una imagen se tuvo que establecer los parámetros para cada dispositivo. Los parámetros se presentan en la tabla 2.

Tabla 2: Tabla de parámetros por equipo.

Nombre de parámetro	Descripción
Porcentaje_w_pantalla, Porcentaje_h_pantalla, Porcentaje_w_dígitos, Porcentaje_h_dígitos,	Para controlar un ancho y alto máximo entre pantalla-dígito, dígitos-dígitos.
Win_size_sauvola, k_sauvo	Parámetros para umbralizacion Sauvola.
Tamanio_led	Tamaño para resize de la pantalla
Tresh_canny_low, Tresh_cannt_high	Umbrales para canny
h_bajo, h_alto, s_bajo, s_alto, v_bajo, v_alto	Valores para segmentación color HSV
Tam_close	Tamaño del kernel de morfología
Delta, min_area, max_area, max_variation	Parámetros del MSER
Corte_pantalla_filas_i, Corte_pantalla_filas_f, Corte_pantalla_columnas_i, Corte_pantalla_ columnas _f	Para recortar la imagen led
Thres_blanco	Parámetro invierte mascara sauvola

Para separar todos los pasos llevados a cabo, en la Figura 2 se muestra un diagrama de bloques para visualizar cada uno de los procesos descriptos en este informe.

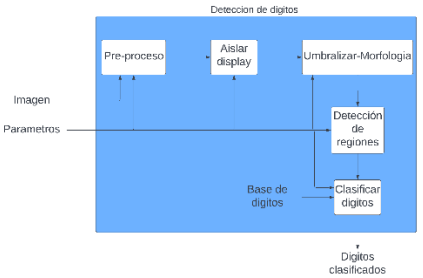


Figura 2: Diagrama de bloques del algoritmo.

A. Pre-proceso:

En el primer bloque denominado pre-proceso, la imagen de entrada se redimensiona. Además, se aplica una máscara color [8], y se utilizan dos metodologías de trabajo distintas según, si el led esta retroiluminado o no.

i. Si la pantalla es retroiluminada:

Al aislar únicamente los dígitos, el interés es poder excluir la pantalla y así redimensionarla poseyendo una tolerancia ante la cercanía o lejanía de la cámara con respecto al equipo. Lo que se aplicó fue una dilatación morfológica [8] para así extender y conectar los dígitos, formando un rectángulo. Luego mediante un bitwise and [8] con la imagen se obtiene la región de interés. En la figura 3 se muestra la salida de este proceso.

Figura 3: Izq: máscara segmentada dilatada. Derecha: pantalla en escala de grises.

ii. Si la pantalla no es retroiluminada:

En este caso, la máscara color segmentada contiene las regiones candidatas a ser la pantalla, por lo tanto, hay que hallar la región correcta. Se aplica un cierre morfológico para mejorar la imagen aislada. En la figura 4 se muestra la salida de este proceso.

Figura 4: Imagen segmentada a color.

B. Aislamiento de led:

En el bloque aislar led se pasa como parámetro las regiones candidatas de pantalla. Se la convierte a escala de grises, se aplica un filtrado gaussiano y se detecta los bordes a través del método de Canny [8], luego se aplica una dilatación para cerrar los contornos. De esto se obtienen los respectivos bordes de las correspondientes áreas de interés, aplicando la función de búsqueda de contornos, se obtienen las áreas cerradas que, al ordenarlas de menor a mayor, la primera que tuviera cuatro vértices sería la pantalla. Tomando los cuatro vértices hallados se emplea una transformación para terminar de aislar la pantalla de la imagen. Se le aplica una redimensión para mantener constante el área de análisis ante cambios en la distancia de captura de la imagen. Por último, se utiliza una corrección gamma [8] a la pantalla en escala de grises para mejorar el contraste del fondo de pantalla con el dígito. En la figura 5 se muestra la salida de este proceso.

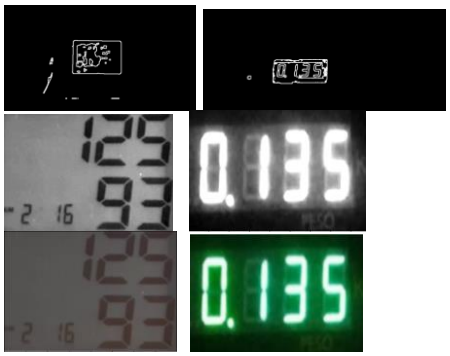


Figura 5: Izquierda de arriba hacia abajo: bordes, led en escala de grises y led color de pantalla no retroiluminado. Derecha de arriba hacia abajo: bordes, led en escala de grises y led color de pantalla retroiluminado.

C. Umbralizado-morfología:

Se utilizan dos metodologías de trabajo distintas según si el led esta retroiluminado o no.

i. Si la pantalla es retroiluminada:

Se realiza una máscara de color sobre la pantalla, binarizando la imagen. En la figura 6 se muestra el resultado de este proceso.

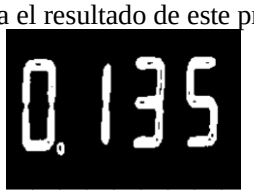


Figura 6: Pantalla umbral izado

ii. Si la pantalla no es retroiluminada:

Se presenta un bajo contraste entre los dígitos de interés y el fondo. Para mejorar esto, se optó por realizar un umbral izado de sauvola [1], el cual calcula un umbral T para cada píxel de la imagen utilizando la siguiente fórmula:

$$T = m(x,y) * ((1+k) * (\frac{s(x,y)}{R} - 1))$$

Donde T es el umbral, $m(x,y)$ es la media local, $s(x,y)$ es la desviación estándar local de una ventana cuadrada. k es un parámetro que pondera el efecto de $s(x,y)$ y R es la desviación estándar máxima. Por último, se emplea un cierre morfológico para mejorar la imagen umbral izada. En la figura 7 se muestra el resultado de este proceso.

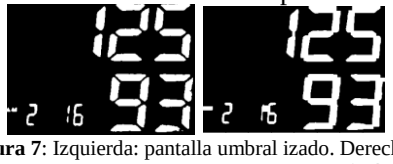


Figura 7: Izquierda: pantalla umbral izado. Derecha: pantalla umbral izado luego de un cierre morfológico.

D. Detección de regiones:

MSER [1] detecta dígitos potenciales al encontrar regiones en las que los valores de intensidad de una región varían mínimamente y cumplen con un área mínima y máxima. No todas las regiones corresponden a dígitos válidos, por lo tanto, se realiza un primer control eliminando las regiones mayores que superen cierta altura y ancho en relación porcentual con el tamaño de la pantalla, esos valores se obtuvieron al medir esas relaciones en el dispositivo físico. En la figura 8 se muestra el resultado de este proceso.

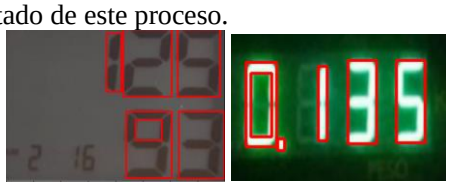


Figura 8: Izquierda: detección de MSER en pantalla no retroiluminado color. Derecha: detección de MSER en pantalla retroiluminado color.

E. Clasificación de dígitos:

Para identificar las regiones que son dígitos, se considera a la primera como un dígito, entonces el resto de estas regiones que no cumplan una relación porcentual con esta primera serán descartadas.

Se utilizaron tres técnicas en la cual cada una etiquetará la región como un dígito entre el cero y nueve. Cada una de estas posee un voto y el digito se clasificará finalmente con el más votado. Las técnicas utilizadas fueron:

i. Se tomaron dígitos base, los cuales se los redimensiona al mismo tamaño que el dígito candidato, para luego ser comparados utilizando el error cuadrático medio donde el que menor error posee es el dígito buscado.

ii. La segunda se inspira en la primera, pero la comparación es mediante la similitud estructural, donde se busca el valor más alto y cercano a uno. Al comparar imágenes, el error cuadrático medio, aunque es fácil de implementar, no es muy indicativo de la similitud percibida. La similitud estructural tiene como objetivo abordar esta deficiencia teniendo en cuenta la textura.

iii. La última toma a las regiones de interés y las divide en los correspondientes siete segmentos. En cada segmento individual, se evaluó su promedio para compararlo con un umbral y si lo supera está encendido. Luego toma la codificación de ceros y unos (diccionario) para identificar al correspondiente dígito por las regiones encendidas o apagadas. En la figura 9 se muestra el aislamiento de los dígitos.

Figura 9: Izquierda: Regiones encontradas como dígitos. Derecha: Dígitos base con los que se compara.

IV. RESULTADOS

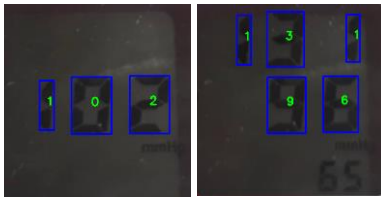
El algoritmo se probó en cuatro tipos de pantalla distintas, cada una con su configuración correspondiente, generalmente encontrando buenos

resultados, los cuales se expondrán a continuación. Se guardaron 158 imágenes extraídas de un video para los tensiómetros y la balanza hogareña. Para la balanza comercial se guardaron 6 fotos, ya que no se tiene acceso al dispositivo.

a. Tensiómetro clásico:

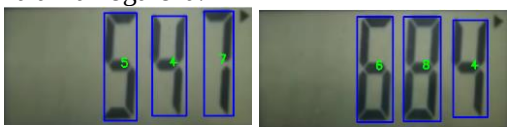
De 158 imágenes, 120 fueron clasificadas correctamente, mientras que el resto presenta fallas. Teniendo un 76 % de precisión.

b. Tensiómetro muñeca:



De 158 imágenes, 150 fueron clasificadas correctamente, mientras que el resto presenta fallas. Teniendo un 95 % de precisión.

c. Balanza hogareña:



De 158 imágenes, 124 fueron clasificadas correctamente, mientras que el resto presenta fallas. Teniendo un 78 % de precisión.

d. Balanza comercial (pantalla retroiluminada):

De las 6 imágenes disponibles, todas fueron clasificadas correctamente.

Cuando las condiciones lumínicas son apropiadas y la captura de la imagen es paralela a la pantalla, se muestran resultados satisfactorios.
Las fallas más comunes del presente algoritmo se presentan cuando la pantalla refleja la fuente de luz, algún objeto sobre el led y cuando la imagen posee demasiada inclinación en cualquier direccion. Por último, se puede presentar una detección incorrecta del dígito, como entre el 0-8,7-3,9-5, por su gran parecido en estos tipos de pantallas.

V. CONCLUSIONES

Como primera aproximación a resolver en este tipo de problemas los resultados que se exhiben son acordes a lo buscado. Los errores que se presentan son por fallas en la detección de la pantalla debido a que no cierra el área de éste por condiciones lumínicas desfavorables, posiciones incorrectas de cámara y fallas al unir los segmentos de la pantalla ya que no lo selecciona como dígito candidato.
Como trabajo futuro se plantea frente a la sensibilidad de la segmentación a color a las condiciones lumínicas para el aislamiento de la pantalla para las pantallas no retroiluminadas utilizar filtros para solventar este problema, por ejemplo, el filtrado retinex. En el caso de las retroiluminadas, mejorar la segmentación de la pantalla ya que con una simple dilatación es insuficiente. Para mejorar la precisión en la clasificación de dígitos se podría hacer uso de técnicas de aprendizaje automático, ya que poseen la ventaja de ser mas robustas por la naturaleza de entrenamiento con un gran volumen de datos de ejemplo.

REFERENCIAS

[1] Finnegan E, Villarroel M, Velardo C, Tarassenko L. Automated method for detecting and reading seven-segment digits from images of blood glucose metres and blood pressure monitors. J Med Eng Technol. 2019 Aug;43.
[2] Tekin E, Coughlan J and Shen H, "Real-time detection and reading of LED/LCD display for visually impaired persons," 2011 IEEE Workshop on Applications of Computer Vision (WACV), 2011, pp. 491-496.
[3] Kanagarathinam K, Sekar K, Text detection and recognition in raw image dataset of seven segment digital energy meter display, Energy Reports, Volume 5, 2019, pp. 842-852
[4] Python Software Foundation. Python Language Reference, versión 3.9. Disponible en https://www.python.org/
[5] Harris C, Millman K, van der Walt S, Gommers R, Virtanen P, Cournapeau D, Oliphant T. Array programming with NumPy. Nature, 2020, pp. 357–362.
[6] Van der Walt S, Schonberger J, Nunez-Iglesias J, Boulogne F, Warner J, Yager N, Yu T. scikit-image: image processing in Python. 2014
[7] Bradski G. The OpenCV Library. Dr. Dobb's; Journal of Software Tools. 2000
[8] Martínez C, Albornoz E, Bugnon L, Ferrante E, *Apuntes de catedra Procesamiento Digital de Imágenes, Ingeniería en Informática, UNL-FICH, 2022.*