Anexo E:

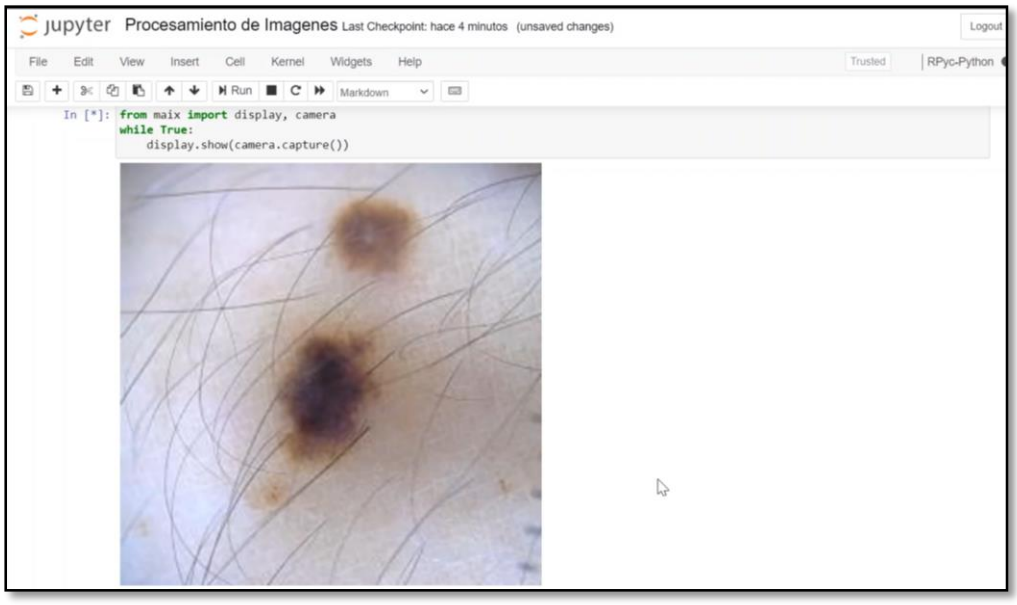
PRUEBAS CAMARA

1. Código para visualizar la imagen en tiempo real

Este código utiliza las bibliotecas “maix.display” y “maix.camera” para implementar un bucle infinito que captura imágenes de la cámara y las muestra en un dispositivo de visualización en tiempo real. La instrucción “camera.capture()” captura un fotograma de la cámara, y “display.show()” presenta ese fotograma en el dispositivo de visualización. El bucle “while True” asegura que este proceso de captura y visualización se repita continuamente, lo que resulta en una transmisión en vivo de la entrada de la cámara en el dispositivo de visualización. Este tipo de código es comúnmente utilizado en aplicaciones que requieren un flujo en tiempo real de imágenes de la cámara, como sistemas de vigilancia, análisis de video en tiempo real o proyectos de visión por computadora.

Figura 1.

Código para visualizar imágenes en tiempo real.



Nota: La figura muestra el código de visualizar imágenes en tiempo real luego de ser ejecutado. Crea-do por los autores.

2. Código para capturar y guardar una imagen

Este código utiliza las bibliotecas “maix.camera” y “maix.display” para capturar una imagen de la cámara y mostrarlas en el dispositivo de visualización. En particular, el método “camera.capture()” captura una imagen de la cámara, y la imagen resultante se guarda en formato JPG en la ruta especificada ("/root/tmp.jpg") mediante el método “img.save()”. Luego, la imagen recién capturada se muestra en el dispositivo de visualización utilizando “display.show(img)”. Este tipo de código es útil para aplicaciones que requieren la captura de imágenes de la cámara en tiempo real y su posterior visualización. La elección entre el formato JPG y PNG para guardar la imagen influye en el tamaño y compresión del archivo, estos formatos son comunes en aplicaciones que manejan grandes cantidades de imágenes.

Figura 2.

Imágenes capturadas en formato "jpg" y "png".

Código para capturar y guardar una imagen

```
#Formato JPG
from maix import camera, display
img = camera.capture()
img.save('/root/tmp.jpg')
display.show(img)
```

```
#Formato PNG
from maix import camera, display
img = camera.capture()
img.save('/root/tmp.png')
display.show(img)
```

Nota: La figura muestra el código de captura de imágenes en formatos "jpg" y "png". Creado por los autores.

3. Código para cambiar el tamaño de la imagen vista en tiempo real

Este código utiliza las bibliotecas "maix.camera", "maix.display" y "maix.image" para configurar la cámara con una resolución de 640x360 píxeles. Luego de ello, capturar y muestra continuamente imágenes en un bucle infinito. Primero, se establece la configuración de la cámara con el tamaño deseado utilizando "camera.config(size=(640, 360))". Luego, en el bucle "while", se captura una imagen con "camera.capture()" y muestra en tiempo real en el dispositivo de visualización mediante "display.show(img)".

Figura 2.

Configuración para cambiar tamaño de imagen con visualización en tiempo real.

Nota: La figura muestra el código para cambiar el tamaño de la imagen mostrada en tiempo real. Creado por los autores.

4. Código para cambiar de la cámara zoom

Este código utiliza las bibliotecas “maix.camera”, “maix.display” y “maix.image” para configurar la cámara con una resolución de 640x360 píxeles, luego captura y muestra continuamente imágenes en un bucle infinito. Además de la captura de imágenes, se aplica corrección de distorsión de lente a cada imagen utilizando el método “lens_corr” de la biblioteca “maix.image”. En este caso, se especifica un parámetro de fuerza (strength) de 1.8 y un zoom de 1.0 para corregir posibles distorsiones de la lente. Este tipo de corrección es común en aplicaciones de visión por computadora donde es crucial obtener imágenes corregidas y nítidas para análisis y procesamiento adicionales. La imagen corregida se muestra en tiempo real en el dispositivo de visualización mediante “display.show(img)”. Este código es útil en entornos integrados o embebidos donde se requiere corrección de distorsión de lente en combinación con la visualización en tiempo real de la salida de la cámara.

Figura 3.

Pasos para corrección de distorsión de lente y zoom de la cámara.

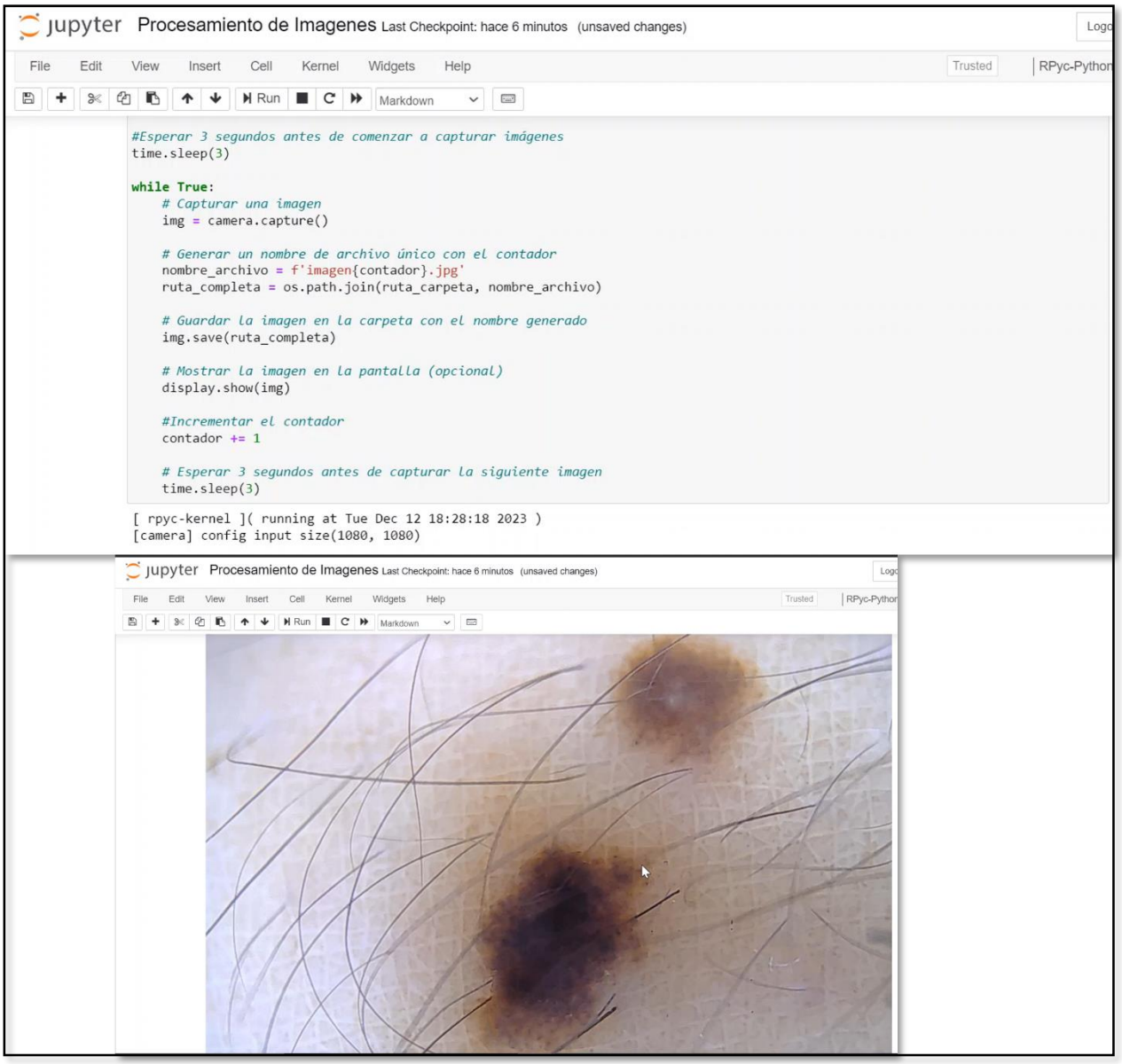
Nota: La figura muestra el código para cambiar el zoom y strength de la imagen mostrada en tiempo real. Creado por los autores.

5. Código para capturar y guardar imágenes de manera continua

Este código utiliza las bibliotecas “maix.camera”, “maix.display”, y otras bibliotecas estándar como os y time para capturar imágenes de una cámara con una resolución de 1080x1080 píxeles en un bucle infinito. Las imágenes capturadas se guardan en una carpeta específica (/root/d) con nombres de archivo únicos que incluyen un contador incremental en el nombre. Además, se muestra cada imagen capturada en tiempo real en un dispositivo de visualización mediante “display.show(img)”. El bucle se repite cada 3 segundos, proporcionando un intervalo entre capturas. Este código es útil para escenarios donde se requiere captura continua de imágenes con nombres de archivo únicos, como en aplicaciones de monitoreo o adquisición de datos.

Figura 4.

Proceso para la captura y almacenamiento continuo de imágenes.



Nota: La figura muestra el código de captura de imágenes luego de ser ejecutado. Creado por los autores.

6. Código para visualización de colores

Este código utiliza las bibliotecas “maix.image”, “maix.display” y “maix.camera” para capturar imágenes desde la cámara en un bucle infinito. Define umbrales de color en el espacio de color Lab para cuatro colores específicos: verde, rojo, amarillo y azul. Luego, utiliza estos umbrales para encontrar y resaltar áreas en la imagen que coincidan con estos colores. Si se detecta un área de color significativa (definida por el ancho y alto de un "blobs"), dibuja un rectángulo alrededor de ella en la imagen y muestra el nombre del color correspondiente. Este código es útil para aplicaciones de procesamiento de imágenes en tiempo real que requieren la detección y resaltado de objetos de colores específicos en una imagen capturada por una cámara.

Figura 5.

Proceso para la visualización de colores en tiempo real.

Nota: La figura muestra el código de visualización de colores en tiempo real de imágenes luego de ser ejecutado. Creado por los autores.