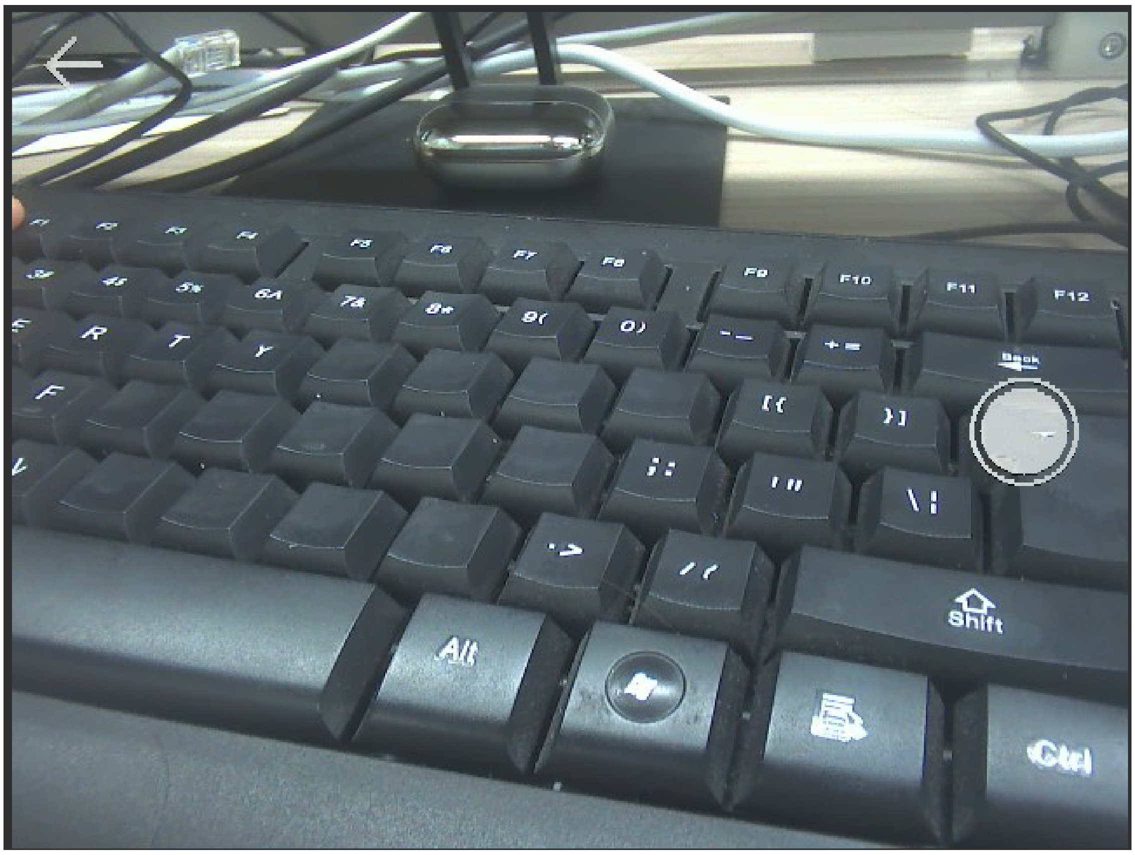
Touch to take a photo

Runtime Results

- Note: This example does not provide an offline version; it must be launched from the pre-installed GUI application.

  The code for this section is located in [\sdcard\apps\camera]

  The run-time results are shown in the image.

  Touch the white button on the right side of the screen to take a photo and save it to the data directory.

Code Explanation

1. Import the necessary modules

```python
import lvgl as lv
import os, gc
import time
import binascii
from ybMain.base_app import BaseApp
from media.media import *
from media.display import *
from machine import TOUCH
from ybUtils.Configuration import Configuration
```

- `lvgl`: A lightweight GUI library for creating graphical user interfaces.
- `os`, `gc`: For file system operations and garbage collection.
- `time`: For time control.
- `binascii`: For generating random file names.
- `ybMain.base_app.BaseApp`: The base application class. Applications inheriting from this class can be integrated into the application manager.
- `media.media` and `media.display`: For screen display and media resource management.
- `machine.TOUCH`: For initializing and operating the touch sensor.
- `ybUtils.Configuration`: For loading configuration files.

2. Initializing the Touch Sensor

```python
tp = TOUCH(0)
```

- `TOUCH(0)`: Initializes the touch sensor on pin 0. `tp` is used to read touch data later.

3. Helper function: Ensure the directory exists

```python
def ensure_dir(directory):
    if not directory or directory == '/':
        return
    directory = directory.rstrip('/')
    try:
        os.stat(directory)
        return
    except OSError:
        print(f"Directory does not exist, creating: {directory}")
        if '/' in directory:
            parent = directory[:directory.rindex('/')]
            if parent and parent != directory:
                ensure_dir(parent)
        try:
            os.mkdir(directory)
        except OSError as e:
            print(f"Error creating directory {directory}: {e}")
            try:
                os.stat(directory)
            except:
                print(f"Still does not exist: {directory}")
    except Exception as e:
        print(f"Unexpected error: {e}")
```

- `ensure_dir()`: Recursively checks and creates a directory to ensure the specified path exists. If the directory does not exist, it attempts to create it.

4. Define the `App` class

```
class App(BaseApp):
    def __init__(self, app_manager):
```

- The `App` class, which inherits from `BaseApp`, is the core of the camera application and is responsible for interface initialization, touch event handling, and camera operations.
- The constructor loads the background icon image (if present) and initializes the configuration and text content.

4.1 Initializing the Interface and Resources

```
def initialize(self):
    self.save_path = "/data/photo/"
    self.prefix = binascii.hexlify(os.urandom(5)).decode()
    self.i = 1
```

- `save_path`: Sets the photo save path to `/data/photo/`.
- `prefix`: Generates a random 5-byte hexadecimal string as the folder prefix.
- `i`: The photo file number, starting at 1 and incrementing.

4.2 Creating the User Interface

```
content = lv.obj(self.screen)
content.set_size(lv.pct(100), 420)
content.align(lv.ALIGN.BOTTOM_MID, 0, 0)
```

- Create the main container `content`, aligning it to the bottom center of the screen, with a width of 100% and a height of 420 pixels.
- Set it to a horizontal layout (`FLEX_FLOW.ROW`), containing the left and right panels.

4.2.1 Left Panel: Text and Buttons

```
left_panel = lv.obj(content)
left_panel.set_size(lv.pct(35), lv.pct(100))
left_panel.set_flex_flow(lv.FLEX_FLOW.COLUMN)
```

- Create the left panel `left_panel`, occupying 35% of the main container's width.
- Add an introductory text label and a save path label, with the text content retrieved from the configuration file.
- Add an "Open Camera" button, which triggers the `open_camera()` method when clicked.

4.2.2 Right Panel: Preview Image

```
right_panel = lv.obj(content)
right_panel.set_size(lv.pct(65), lv.pct(100))
img = lv.img(right_panel)
```

- Create the right panel `right_panel`, occupying 65% of the main container's width.
- Load and display the camera preview (if present), center-aligning it and maintaining its original aspect ratio.

5. Camera Preview and Photo Capture

5.1 Opening the Camera Preview

```
def open_camera(self, event):
    gc.collect()
    img2 = image.Image(640, 480, image.RGB565)
    img2.clear()
```

- Call the garbage collector to free up memory.
- Create an image `img2` with a resolution of 640x480 and clear its contents.
- Draw the capture button (the circle in the middle right corner) and the return button (the arrow in the upper left corner) on the image.

```
Display.show_image(img2, 0, 0, Display.LAYER_OSD0)
```

- Display the drawn interface on the screen (Layer `LAYER_OSD0`).

5.2 Camera Preview Main Loop

```
while True:
    point = tp.read(1)
    if len(point):
        pt = point[0]
        if pt.event == TOUCH.EVENT_DOWN:
            if pt.x > 520 and pt.y > 150 and pt.x < 640 and pt.y < 300:
                self.take_picture(img)
                img.draw_rectangle(4, 4, 640 - 8, 480 - 8, (200, 0, 0),
thickness=8)
                Display.show_image(img, 0, 0, Display.LAYER_OSD3)
                time.sleep_ms(100)
            elif pt.x < 60 and pt.y < 60:
                self.exit_demo()
                time.sleep_ms(10)
                break

    img = self.pl.sensor.snapshot(chn=CAM_CHN_ID_1)
    Display.show_image(img, 0, 0, Display.LAYER_OSD3)
    time.sleep_ms(10)
```

- Continuously read touch data:
  - If the touch point is in the right capture button area (x > 520, y > 150, x < 640, y < 300), call `take_picture()` to take a picture and display a red border as feedback.

    - If the touch point is in the upper-left return area (x < 60, y < 60), call `exit_demo()` to exit camera mode.
- Use `snapshot()` to obtain the camera preview image and display it on the screen in real time (`LAYER_OSD3` layer).
- Update every 10 milliseconds.

5.3 Taking and Saving Photos

```python
def take_picture(self, img):
    ensure_dir(self.save_path + str(self.prefix) + "/")
    path = self.save_path + str(self.prefix) + "/" + str(self.i) + ".jpg"
    self.i += 1
    img.save(path)
    time.sleep_ms(2)
```

- Call `ensure_dir()` to ensure the save path exists.
- Generate the save path and file name (format `{save_path}/{prefix}/{i}.jpg`) and increment the file number.
- Save the current preview image as a JPEG file.

6. Exit Camera Mode

```python
def deinitialize(self):
    img2 = image.Image(640, 480, image.RGB565)
    img2.clear()
    Display.show_image(img2, 0, 0, Display.LAYER_OSD3)

def exit_demo(self):
    self.deinitialize()
    return
```

- `deinitialize()`: Creates a blank image and displays it in the `LAYER_OSD3` layer, clearing the camera preview.
- `exit_demo()`: Calls `deinitialize()` to clean up resources and exit camera mode.