Display and touch coordinates system

When using the Lilygo T-Display S3 cap with libraries like TouchLib and TFT_eSPI, rotating the touch area's coordinates is currently not possible. However, if you're using TFT_eSPI as the screen driver, you can manually adjust the touch coordinates to match the screen's orientation. This ensures that touch input aligns correctly with what's displayed. The following picture shows the direction of the touch coordinates:

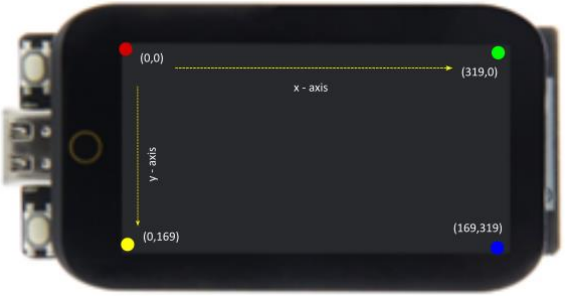
picture 1/ touch area coordinates

Screen rotation by TFT_eSPI library and screen coordinates

picture 2/ tft.setRotation(0)

picture 3/ tft.setRotation(1)

picture 4/ tft.setRotation(2)



picture 5/ tft.setRotation(3)

When designing the screen interface, such as buttons, it's important to consider the screen's rotation settings. The coordinates used for designing the screen elements should be based on the screen's rotated coordinates. However, it's essential to note that the coordinate system used for the touch function remains unaffected. The touch function will still use the unrotated coordinates regardless of the screen rotation settings.

Only in case the screen is not rotated using the **tft.setRotation(0)** function, the touch function and the screen coordinates will align properly. However, if the screen is rotated using a different rotation value, the touch function will continue to use the unrotated coordinates.

To ensure that the touch input aligns correctly with the screen elements (such as button areas), the coordinates need to be transformed accordingly. This means through the transformation touch coordinates fit with the widget coordinates. This means the touch event will only be processed when touching the widget area, here the button area.

By considering the rotation settings during the design process and transforming the coordinates as needed, you can ensure that the screen elements and touch input work seamlessly together. Regardless of the screen's rotation configuration, this approach will help ensure that the touch input accurately corresponds to the intended screen elements.

Samples

Botton on screen in landscape (tft.setRotation(1))

The button will be drawn by the tft_eSPI library, the touch function will be handled by the use of TouchLib library.

The parameters for the botton are: (x=220, y=120, w=80, h=40). The script will switch a LED on pin 4 and also a symbol on the screen. The label in the button will be changed from “off” to “on”. After downloading the script (ui_sample_2) the screen looks like:

Touching on the button will change the screen like following:

The button label changed from “off” to “on” and the LED symbol from “red” to “green”. In addition the coordinates of the touch t.x and t.y will be displayed. When touching on some other screen area outside the button area only the touch coordinates will be changed. So a change of the button label and the LED color will only happen as long as the touch is inside the button area.

The necessary transformation of the coordinates can be seen under void transform_coordinates(int option). With the given button parameters the screen can be properly displayed with tft.setRotation(1) and tft.setRotation(3). With the other rotation parameters 0 and 2 the button would be outside the screen area, because for drawing the button, the coordinates of the screen coordinates are the one to consider (see above pictures). So also the touch function can not be operated.

In case the button is located in an area which is visible for all for rotations, also the touch function can work for all rotation parameters (ui_sample_1). Caution: base of the screen design is the coordinate system of the screen not the touch (see picture 2 to 5).

The two samples are very simple and for understanding the touch issue with the given libraries. Also the design is simple. Very nice looking designs are possible by the use of the LVGL library. Also many nice widgets are offered and don`t have to be programed. Anyhow the library is very complex and not easy to understand. No touch support for the Lilygo T-Display S3 cap is currently available. This makes the programing and the use of the Lilygo display even more complex. Some sample sketches will follow.