Circuit Cellar DesignStellaris 2010

An Eye Tracking System using Toy Camera

Abstract

TI2847

23 June 2010
An Eye Tracking System using Toy Camera

An eye tracking system measures the point of gaze by analyzing eye movement. It is used in many fields including commercial services. In this project, the basic eye tracking system was developed with the Stellaris processor board and a USB toy camera.

![System Overview](image_url)
Eye tracking systems use the relation between the corneal reflection, and the center of the pupil to identify gaze direction. It uses infrared light to distinguish pupil shape from iris. A toy camera (Che-ez! spyz) has been altered to capture infrared vision. Two infrared LEDs attached have sharp directional beams. The EKK-LM3S9B96 evaluation board was used without expansion.

Figure 2: Schematic
The Stellaris MCU works as a USB host, and controls the camera. The user-button on the board switches the calibration/tracking modes. In the calibration mode, the captured eye image with detected markers is displayed on the browser screen of web clients. In the tracking mode, gaze points are plotted on a white image. The gaze point detector finds the bounding rectangles and centers of both corneal reflection and pupil to calculate the gazing point. The HTTPD generates a HTML and transmits JPEG images. This HTML contains a JavaScript and refreshes the image four times per second.

The head should be kept still during eye tracking. You will see the detected markers over eyes in the captured view. Switch to the plot mode, and gaze at a corner of the screen for a while. The plots will gradually move to that direction.

Although the eye tracking was barely achieved with this system, its performance falls far short of that of the commercial products. However, this experiment will be a good step to design a practical eye tracking system.
The code snippet of the image capture sequence is shown below.

```c
void camera_start( unsigned long ulBulkInPipe )
{
    // activate bulk transfer
    if( usb_set_altinterface(1)!=0 ) { ... 
        xTaskDelay( 2000 );
    }

    // clear all error and data
    if( usb_control_msg( USB_RTYPE_DIR_OUT, CMDID_CLEAR_COMMS_ERROR, ... 

    // set SDRAM entry pointers
    if( usb_control_msg( USB_RTYPE_DIR_OUT, CMDID_SET_IMAGE_INDEX, ... 
        if( usb_control_msg( USB_RTYPE_DIR_IN, CMDID_GET_TABLE_ENTRY, ... 

    // initialize gaze point detector
    trace_init( data, w8, HGT );

    for( ;; ) {
        USBHCDMain();

        // grab image
        if( usb_control_msg( USB_RTYPE_DIR_OUT, CMDID_GRAB_IMAGE, ... 

        // detect gaze point while processing the grab
        trace_gazepoint();

        // wait until grabbing is done
        do { ... 
        } while( errinf.error==CAMERR_BUSY );

        // retrieve band data (twice)
        for( i=0; i<HGN; i++ ) {
            USBHCDMain();

            // request band data
            if( usb_control_msg( USB_RTYPE_DIR_OUT, CMDID_UPLOAD_SDRAM, ... 

            // obtain band data
            if( USBHCDPipeRead( ulBulkInPipe, dptr, sdramex[i].size ) ... 

            // retry transfer - it frequently fails if the data size
            // is not the multiples of a strip size (1288 bytes).
            if( usb_control_msg( USB_RTYPE_DIR_IN, CMDID_GET_LAST_ERROR, ... 
                l--; } 
                continue;
            }

            // reform band data - exploit 'R' pixels
            ... // mirroring – flip horizontally
            ...
        }
    }
}
```

List 2: Image Capture Sequence