Coloured-Object Tracking Camera

ECE 492 Group 4 Project Design

Group members

Ryan Corpuz

- Servo control
  - Custom PWM
  - Rotational velocity variations

Hang Peng

- Camera and monitor interfacing
  - Video input signal format conversion
  - Threshold components

Jingjing Liang

- Algorithms
  - Positioning
  - Pixel Comparison

Functionality/Motivation

Vincent Lee, “Jank Edit 2.0”
https://www.youtube.com/watch?v=jfmxrR4WlBg

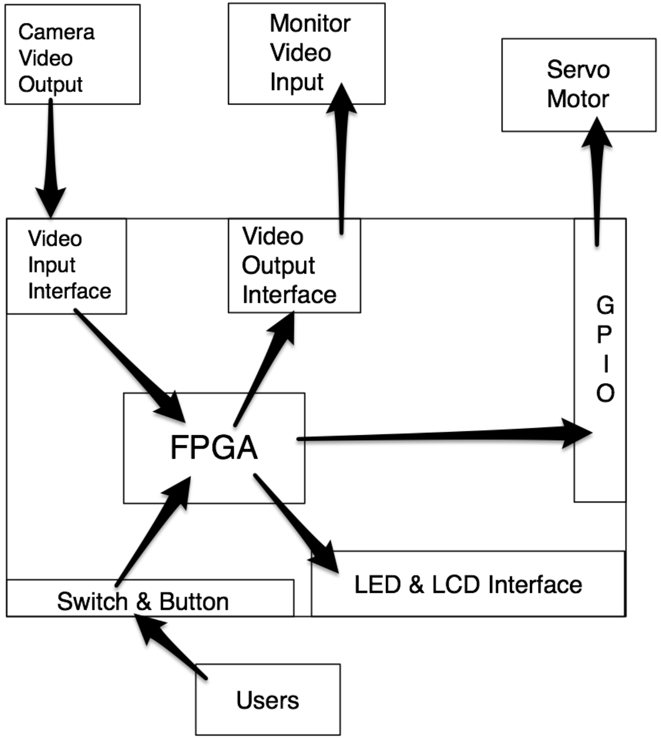
MOTIVATION

- Original idea came from this video
  - Distance as a safety factor but prevents constant view of target
- Can be used for many other things
  - Security
  - Don’t have a camera man
  - Tracking images in unsafe environments

FUNCTIONALITY

- Input images via camera input
- Threshold image for specified colour being tracked
- Calculate centroid of the object and it’s position with respect to the camera’s center (center of image)
- Output appropriate signals to servos to orient camera such that the center of the object is at the center of the camera’s view
- Output camera images to a monitor via VGA port

Hardware Design

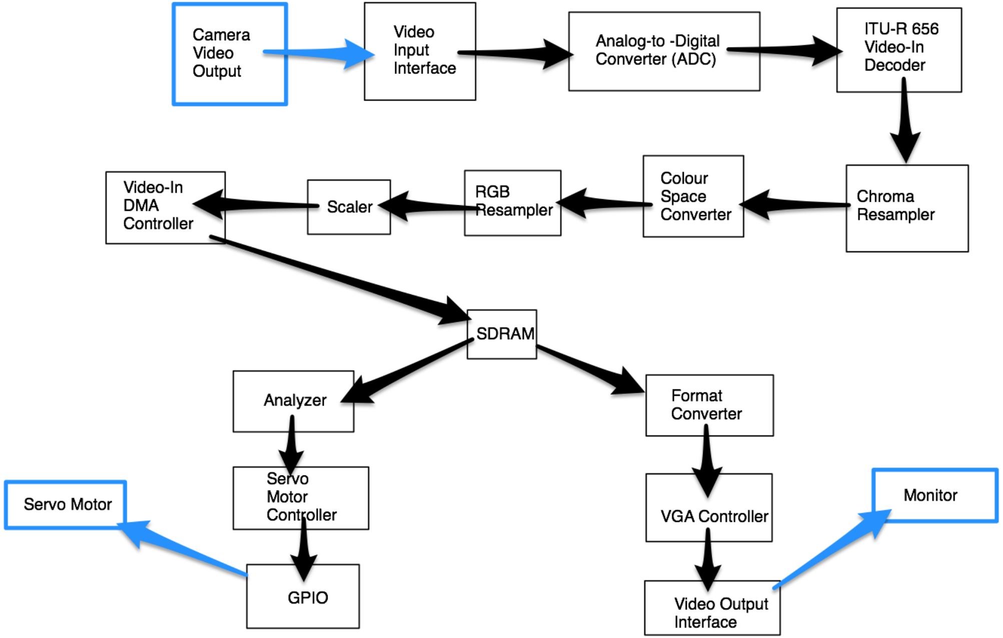


- Video Processing:
  - NTSC input signal --- FPGA
  - FPGA --- RGB output signal

- User Interfacing:
  - Threshold value control (Buttons)
  - Operations indication (LCD)

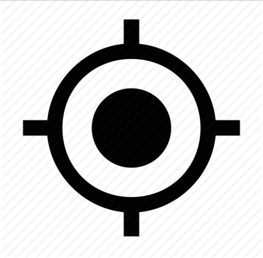
- Servo Motor Control:
  - Custom PWM

Data Flow



Altera Corporation. ***University Program IP Cores.*** **Video IP Suit**

Software Design

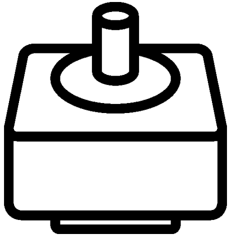


1. Threshold Value Comparison

   Input: RGB video signal
   Output: Coordinate of center of the target

2. Calculate Direction of Object and Displacement

   Input: Coordinates of the centroid of the frame
   Output: PWM instructions



3. Generate PWM for rotation

   Input: PWM instruction structure
   Output: GPIO control signals

1.http://www.psd100.com/stepper-motor-icon-2/#.VNw6ZUKqu5I 2.http://www.iconpng.com/icon/48480 3.http://www.iconpng.com/icon/24323

Challenges

- Time constraint
    - Minimize computation time for each iteration to maximize FPS
- Smoother Panning
    - Acceleration/Velocity changes with respect to object’s displacement
    - Dependent on FPS

Component Example

SERVO PWM

```
if ( current_state = high ) then
    if ( pulse_count > 0 ) then
        pulse_count := pulse_count - 1;
        coe_servo <= '1';
    elsif ( pulse_count = 0 ) then
        current_state <= low;
        case direction(7 downto 0) is
            when "00000000" => pulse_count := NEUTRAL;
            when "00001111" => pulse_count := CW;
            when "11111111" => pulse_count := CCW;
            when others => pulse_count := NEUTRAL;
        end case;
    end if;
elsif ( current_state = low ) then
    if ( period_count > 0 ) then
        period_count := period_count - 1;
        coe_servo <= '0';
    elsif ( period_count = 0 ) then
            period_count := REFRESH;
            current_state <= high;
    end if;
end if;
```

Code Example

```
/*Threshold comparison pseudo code */
SET Row to 320
SET Column to 240
SET Threshold_range to 30
SET Threshold to [255,0,0]
INIT row_index to zero
INIT column_index to zero
INIT current_addreess
INIT output[Row][Column]

WHILE row_index is less than Row THEN
  WHILE column_index is less than Column THEN
    SET current_address to Address[current pixel]
    GET [R,G,B] FROM current_address
    COMPUTE  difference FROM Threshold and [R,G,B]
    IF difference <  Threshould_range THEN
      output[row_index][colume_index] = 1
    ELSE output[row_index][colume_index] = 0
    ENDIF
    colume_index++
  ENDWHILE
  row_index++
ENDWHILE
RETURN output
```

```
/*positioning pseudo code*/
INIT counter, x_start, x_end, y_start, y_end to ZERO
INIT x,y,x_temp,,y_temp to ZERO
INIT centre [0,0]
WHILE row_index is less than Row THEN
  WHILE column_index is less than Column THEN
    IF output[row_index][column_index] EQUAL 1 THEN
      SET y_start to column_index
      WHILE output[row_index][column_index] EQUAL 1 THEN
        counter ++
        column_index ++
        SET y_end to column_index
      ENDWHILE
      GET y distance
      SET y_temp to y distance
      IF y_temp > y
          SET y to y_temp
      ENDIF
   ENDIF
   cloumn_index++
  ENDWHILE
  row_index++
ENDWHILE
```

```
WHILE column_index is less than Row THEN
  WHILE row_index is less than Column THEN
    IF output[row_index][column_index] EQUAL 1 THEN
      SET x_start to row_index
      WHILE output[row_index][column_index] EQUAL 1 THEN
        counter ++
        row_index ++
        SET x_end to row_index
      ENDWHILE
      GET x distance
      SET x_temp to x distance
      IF x_temp > x
          SET x to x_temp
      ENDIF
   ENDIF
   row_index++
  ENDWHILE
  column_index++
ENDWHILE
RETURN [x,y]
/* displacement calculation pseudo code*/
RETURN [x-160, y-120]
/* Then use the vector to generate PWM*/
```

Test Plan

- Threshold testing
  - Figure out appropriate threshold ranges for the colour
- Stationary tracking test
  - Outputting object displacement (x,y)
- Servo testing
  - Test rotational velocity and acceleration with respect to various supplied voltages and input signals
  - Appropriate motion with object displacements (rotational velocities)
- Output camera data to monitor
  - Display camera image and threshold image on monitor

Future Work

- Custom Settings
  - Offsetting tracked object
  - Boundary threshold
  - Panning threshold
- Minimize form factor

Questions?

Thanks for Watching