Functional requirements

- Avoids obstacles effectively.

- Measures positions of balls.

- Transmits ball information to control subsystem.

Non-functional requirements

- Can collect position and colour of balls given that:

1. the distance is within 20cm - 80cm

2. the angle is less than 45 degrees

3. the ball does not touch the edges of the frame

- In addition, balls can be detected (but data cannot be recorded) within:

1. distance greater than 14cm

2. angle smaller than 50 degrees

- Can identify colours of provided balls (red, pink, yellow, green, blue) under sunlight.

- Can accurately determine distance to balls under sunlight and lamp light.

- Automatic gain and exposure calibration to adapt to different light levels.

\*- Accelerometer data used to calculate inclination of rover.

- Accelerometer and camera can be calibrated remotely.

- Processes data 6 times per second.

- UART uses acknowledgement to ensure:

1. all messages are transmitted successfully

2. important messages are transmitted correctly

- UART uses a ‘loose’ handshake when transmitting data, meaning processor can process data, rather than sleep, when waiting for responses. This was chosen over a ‘firm’ handshake because this method is much more resilient against errant characters or missing characters in transmission.

- Uses dynamic arrays to allow system to easily scale to any number of balls if necessary.

Testing conditions

- Tested under sunlight(6500K?) and 3000K 5W LED lamp.

1. Varied brightness by varying distance to lamp or closing blinds partially.

2. Varied direction of lamp and sunlight by rotating camera and balls.

- Background was matte painted wall with wooden floor.

Information flow in ball detection

The image processor receives a stream of RGB pixel values from the camera, this data could have been converted to HSV, however, RGB was used rather than HSV because the hue varied greatly with lighting conditions, which led to poor ball detection and obstacle avoidance using HSV. Using HSV would have also required single cycle division in hardware, which would be expensive in terms of chip area.

The image processor uses 'filters' to determine whether a given pixel could belong to a ball. There are two sets of these filters. The first set (ball detection filters) was designed to reduce the reporting of false negatives, across a wide range of lighting conditions, however, this set provides no data about colour. For colour detection, a second set of filters (colour filters) was designed, this set contains five filters, one for each ball colour. The colour filters have a very low chance of reporting false positives under sunlight, but they are very inaccurate under other lighting conditions.

Data from the filters is accessible to the NIOS II processor through a FIFO message buffer. This includes words containing the bounds of any balls detected, as well as the bounds of each of the five colour filters.

The NIOS II processor removes erroneous data from the detected balls, then compares the data with the bounds of the colour filters, this gives the positions and colours of all visible balls. If a ball is in the path of the rover, and the rover is currently moving, a signal is transmitted over UART to stop the drive subsystem, along with data of the problematic ball. Otherwise, if a new ball is visible, but has not been seen before, its data will be transmitted to the command subsystem.

Information flow with inclination calculation

The accelerometer data is transmitted to the NIOS II processor over SPI, the y-axis and z-axis readings are used to calculate the inclination angle of the rover. This requires the arctan function, a fixed point approximation was created for this using the first three terms of the Taylor series. This approximation is only accurate for inclination between +-20 degrees.

Calibration

The gain and exposure can be automatically adjusted when there are no balls in the field of view. The calibration system first sets both parameters to their maximum value, then reduces them until none of the colour filters are triggered and no balls are detected. This calibration process can be triggered remotely over UART.

UART Interface to control subsystem

The UART interface allows information to be transmitted according to the following rules:

If there is a ball within 30 cm of the camera, an emergency stop occurs.

If a new ball is found, drive is paused to gather data.

Ball data is transmitted after stopping or pausing drive, if an emergency stop occurred, the first batch of data describes the ball causing the emergency stop.

If no new ball data is left to transmit, the continue signal is transmitted, this allows drive to continue moving, or in the case of an emergency stop, informs command that a ball was in the path of the rover.

The control subsystem informs the FPGA whenever drive starts or stops moving, this ensures that drive has received stop commands correctly. Control also sends acknowledgements for each set of ball data, while this does not prevent errors, it does ensure that ball data is transmitted. Calibration and reset can also be triggered remotely by the command subsystem.

While the UART is not being used for other processes, the inclination of the rover is transmitted.

Vision ---> Control

s - Emergency Stop

(If stopping reason unknown, send {bX00+00} after stop, otherwise send data from ball causing stop)

p - Pause to gather data

c - Continue after gathering data

b - Ball data {'b', colour, 2 digit normal distance, sign, 2 digit perpendicular distance}

( +right, -left) (R,P,Y,G,B,U - Colours)

t - Tilt data {'t', sign, 2-digit inclination}

Control ---> Vision

m - Movement command received

s - Stop command received

a - Acknowledgement, sent after each set of ball data

R - Reset all variables, trigger autofocus

C - Calibrate gain

FPGA resource usage

Flow Status Successful - Tue Jun 15 15:37:23 2021

Quartus Prime Version 16.0.0 Build 211 04/27/2016 SJ Lite Edition

Revision Name DE10\_LITE\_D8M\_VIP

Top-level Entity Name DE10\_LITE\_D8M\_VIP

Family MAX 10

Device 10M50DAF484C7G

Timing Models Final

Total logic elements 12,204 / 49,760 ( 25 % )

Total combinational functions 10,595 / 49,760 ( 21 % )

Dedicated logic registers 7,474 / 49,760 ( 15 % )

Total registers 7541

Total pins 171 / 360 ( 48 % )

Total virtual pins 0

Total memory bits 1,319,096 / 1,677,312 ( 79 % )

Embedded Multiplier 9-bit elements 6 / 288 ( 2 % )

Total PLLs 1 / 4 ( 25 % )

UFM blocks 0 / 1 ( 0 % )

ADC blocks 0 / 2 ( 0 % )

Estimated Total logic elements 13,020

Total combinational functions 10569

Logic element usage by number of LUT inputs

-- 4 input functions 5418

-- 3 input functions 2869

-- <=2 input functions 2282

Logic elements by mode

-- normal mode 9093

-- arithmetic mode 1476

Total registers 7581

-- Dedicated logic registers 7581

-- I/O registers 0

I/O pins 171

Total memory bits 1319096

Embedded Multiplier 9-bit elements 6

Total PLLs 1

-- PLLs 1

Maximum fan-out node Qsys:u0|Qsys\_altpll\_0:altpll\_0|Qsys\_altpll\_0\_altpll\_u3t2:sd1|wire\_pll7\_clk[0]

Maximum fan-out 4232

Total fan-out 72189

Average fan-out 3.77