Algorithm Implementation for Advanced Driver Assistance Systems

M. Sc. Fynn Schwiegelshohn, Michael Hübner
Club Vivado 2015
Motivation

• Attention while driving
  – Every 8th accident occurs due to inattentiveness of one of the involved drivers
  – 16% of severe traffic accidents with trucks involved lead back to drowsy truck drivers
  – Accidents where the cause is drowsiness happen during nighttime

• State of the art
  – Monitoring of several parameters such as steering behavior
  – Monitoring the vehicles position on the street
  – Monitoring the driver through a camera
Experimental Setup and Initial Algorithm

Kinect

USB

ZedBoard

Ethernet

Cockpit (Simulator)

Image acquisition

Find face

[Face found]

Find eyes

[Face not found]

[Eyes not found]

[Eyes found]

Generate warning
Algorithm - Complete

Initialisation of System

Image acquisition

Find face

Find eyes

Warning generation

Dangerous situation?

Generate warning

[Foreground isolation & face tracking]

Generate warnings based on hysteresis results

No face found

No face found

[No Eyes found]

[No danger]

No warning generation

[Face found]

[No danger]

[No danger]

[Danger]

[No Eyes found]
Foreground Isolation

- Acquire depth image
- Binary thresholding
- Canny-Edge detection
- Contour detection
- Interpret largest contour as foreground region
- Project the foreground region on the RGB/IR image
- Execute face detection only in the foreground area
Acceleration with Foreground Isolation

- Comparison is done with images as input
- Negative detections require more processing time than positive
- Processing time per image is compared

<table>
<thead>
<tr>
<th></th>
<th>Comparison to base algorithm ( \Phi of \Delta t ) (positive)</th>
<th>Comparison to base algorithm ( \Phi of \Delta t ) (negative)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base algorithm</td>
<td>100 %</td>
<td>100 %</td>
</tr>
<tr>
<td>Foreground isolation (software)</td>
<td>91 %</td>
<td>40 %</td>
</tr>
<tr>
<td>Foreground isolation (hardware)</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

- What factor is achievable with hardware acceleration?
Hardware Foreground Isolation (1)

- Vivado HLS for accelerator design
  - Using available HLS functions for preprocessing
- Problem: Depth image is 16UC1
  - Cannot be defined as parameter in Vivado HLS
  - 16UC1 is converted to 8UC3 (sw) and then to 8UC1 (hw)
Hardware Foreground Isolation (2)

• Calculate angle out of both sobel images
  – First approach: Use atan2 function (hls_math.h)
    
    ```c
    hls::atan2(sobely.val[0], sobelx.val[0]);
    ```

• Problem: Utilization estimates are too high

<table>
<thead>
<tr>
<th>Utilization (%)</th>
<th>7</th>
<th>35</th>
<th>84</th>
<th>291</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>BRAM_18K</td>
<td>DSP48E</td>
<td>FF</td>
<td>LUT</td>
</tr>
</tbody>
</table>

– Second approach: Approximate atan2

```c
// approximation of angle = atan2(sobely.val[0], sobelx.val[0]) + PI_2;
  r = (sobelx.val[0] - y) / (sobelx.val[0] + y);
  angle = PI_4 - PI_4 * r;
```

<table>
<thead>
<tr>
<th>Utilization (%)</th>
<th>7</th>
<th>39</th>
<th>14</th>
<th>51</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>BRAM_18K</td>
<td>DSP48E</td>
<td>FF</td>
<td>LUT</td>
</tr>
</tbody>
</table>

Hardware Foreground Isolation (3)

• Non-maximum Suppression
  – Cannot be handled as a data stream
  • Past data and future data needs to be available for current pixel
  – Implementation of line buffers necessary

```
// fill lineBuffer and load filterMask
LineBufferLoop: for (int row=0; row<rows+1; row++)
{
  for(int col=0; col<cols+1; col++)
  {
    if(col < cols & row < rows)
    {
      absvalueMat >> lineBuffer[2][col];
    }
    if(col < cols)
    {
      lineBuffer[0][col] = lineBuffer[1][col];
      lineBuffer[1][col] = lineBuffer[2][col];
    }
  }
}
```

```
Line Buffers
```

```
Filter Mask
```

- 135° 90° 45° ...
- 0° 0° ...
- 45° 90° 135° ...
Hardware Foreground Isolation (4)

- Fill the filter mask with the relevant line buffer data
- Perform angle comparison just like in C/C++
- Save result in new image Matrix

```c
for(int i=0; i<3; i++)
    
#pragma HLS unroll
    filterMask[1][0] = filterMask[1][1];
    filterMask[1][1] = filterMask[1][2];
    filterMask[1][2] = lineBuffer[i][col].val[0];

    if(col < cols && row < rows)
        angleMat >> angle;
        angle_value = angle.val[0];

    if(angle_value == angle_0)
        if(filterMask[1][1] >= filterMask[0][1] && filterMask[1][1] >= filterMask[2][1])
            gradient_value = filterMask[1][1];
        else
            gradient_value = 0;
    else if(angle_value == angle_45)
        if(filterMask[1][1] >= filterMask[2][2] && filterMask[1][1] >= filterMask[0][0])
            gradient_value = filterMask[1][1];
        else
            gradient_value = 0;
    else if(angle_value == angle_90)
        if(filterMask[1][1] >= filterMask[1][2] && filterMask[1][1] >= filterMask[1][0])
            gradient_value = filterMask[1][1];
        else
            gradient_value = 0;
    else if(angle_value == angle_135)
        if(filterMask[1][1] >= filterMask[0][2] && filterMask[1][1] >= filterMask[2][0])
            gradient_value = filterMask[1][1];
        else
            gradient_value = 0;
    else
        gradient_value = 127;

    if(col>0 & row>0)
        msumpressionMat << gradient_value;
```
IP-core Integration in System

- Two IP-cores
  - toGray + VDMA
  - toThreshold + VDMA
- Connected to PS over AXI-Interconnect
Chip Utilization after Implementation

Utilization (%)

- MMCM
- BUFG
- DSP48
- BRAM
- I/O
- Memory LUT
- LUT
- FF

Utilization (%)
Utilization of the Chip – Device View

- toGray IP-core + VDMA (purple)
- AXI Interconnect (yellow)
- Foreground Isolation IP-core + VDMA (orange)
- Reference Design (rest)
Functional Verification

- Comparison is done with images as input
- Negative detections require more processing time than positive
- Processing time per image is compared

<table>
<thead>
<tr>
<th></th>
<th>Comparison to base algorithm</th>
<th>Comparison to base algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ø of Δt (positive)</td>
<td>Ø of Δt (negative)</td>
</tr>
<tr>
<td>Base algorithm</td>
<td>100 %</td>
<td>100 %</td>
</tr>
<tr>
<td>Foreground isolation (software)</td>
<td>91 %</td>
<td>40 %</td>
</tr>
<tr>
<td>Foreground isolation (hardware)</td>
<td>48 %</td>
<td>10 %</td>
</tr>
</tbody>
</table>
Actual Use Case Evaluation

Movement of the driver and comparison between detected and actual state

- Correct position
- Movement towards edge of frame
- Head turns side ways
- Eyes closed

Eye state
Eye detection
Face state
Face detection

Frame
Computation time on the ZedBoard

Average Processing time per frame

<table>
<thead>
<tr>
<th></th>
<th>Positive detection</th>
<th>Negative Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi$ of $\Delta t$</td>
<td>0,46 s</td>
<td>2,61 s</td>
</tr>
</tbody>
</table>

Average reaction time before warning is generated

<table>
<thead>
<tr>
<th></th>
<th>Rising edge</th>
<th>Falling Edge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face</td>
<td>0,92 s - 3,53 s</td>
<td>5,22 s - 5,68 s</td>
</tr>
<tr>
<td>Eyes</td>
<td>1,38 s - 1,84 s</td>
<td>1,38 s - 1,84 s</td>
</tr>
</tbody>
</table>

- Real performance does not represent the initial test case results
- The driver of the Kinect does not allow direct access to the allocated memory for the respective images
  - Several copy commands have to be executed
  - Severe performance bottleneck
4. Conclusion

- Robust face and eye detection
- System is usable during day and night time (IR-image)
  - Kinect provides RGB and IR-image
- Hardware Acceleration of System improves performance
  - Only when testing the performance of the hardware not the complete setup
  - Actual use case performance with hardware acceleration is not significantly better compared to only-software implementation
- Currently, no possibility to detect increasing drowsiness of the driver
4. Outlook

- Accelerate face detection with FPGA
  - Using Vivado HLS to generate IP-cores with the OpenCV algorithms

- Find a workaround for the Kinect memory issue

- Reaction times have to be improved (especially when detecting faces)

- Long term drowsiness detection
Thank you for your attention!