<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Wide field of view (WFoV) imaging for consumer devices</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author(s)</strong></td>
<td>Corcoran, Peter</td>
</tr>
<tr>
<td><strong>Publication Date</strong></td>
<td>2014</td>
</tr>
<tr>
<td><strong>Link to publisher's version</strong></td>
<td><a href="http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6775955">http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6775955</a></td>
</tr>
<tr>
<td><strong>Item record</strong></td>
<td><a href="http://hdl.handle.net/10379/5573">http://hdl.handle.net/10379/5573</a></td>
</tr>
<tr>
<td><strong>DOI</strong></td>
<td><a href="http://dx.doi.org/10.1109/ICCE.2014.6775955">http://dx.doi.org/10.1109/ICCE.2014.6775955</a></td>
</tr>
</tbody>
</table>
Wide Field of View Imaging for CE Devices

Peter Corcoran,
Nat. Univ. Ireland Galway
The Problem

Dynamic remapping of WFoV sensor data into conventional image frame ...
The Problem to be Solved? (1)
Or given this ...
How can I get this?
And squeeze into CE Device!
The Solution
Smart Geometrical Distortion Unit Overview
SGDU Overview

- Hardware IP core solution that can apply geometric transformations on images/video frames in real time

- Grids can be static and dynamically configured (frame by frame), with capability to combine static with dynamic generated grids
  - e.g. sensor tilt correction (static grid) + lens barrel distortion (static) + every frame rotation in-plane correction (dynamic VIS analysis module)

- Any geometrical transformation can be implemented
  - Lens distortion correction
  - Perspective correction
  - Affine transformations

- Low memory bandwidth required
Mapping the Image Frame
From sensor to output ...
Is not easy ...
Tiles are not regular ...  

Coordinates found by the Edge Tracer

Each of these pixels must be the center of a 4x4 area

Green area is added by the Tile Border Extender
Lines are not straight ...

Start  \( P0 (x_0, y_0) \)  \( \rightarrow \)  \( P1 (x_1, y_1) \)

Step 1  \( P0 (x_0, y_0) \)  \( \rightarrow \)  \( P1 (x_1, y_1) \)

Step 2  \( P0 (x_0, y_0) \)  \( \rightarrow \)  \( P1 (x_1, y_1) \)

Step 3  \( P0 (x_0, y_0) \)  \( \rightarrow \)  \( P1 (x_1, y_1) \)

...  \( \rightarrow \)  \( P1 (x_1, y_1) \)

Step 11  \( P0 (x_0, y_0) \)  \( \rightarrow \)  \( P1 (x_1, y_1) \)
System Architecture

Hardware pipeline generates output image frame.
Top Level Architecture

- Input Image
- DRAM
- Output Image
- GDE
- Geometrical Distortion Core(s)
- Distortion Formatter
- Output Formatter(s)
- CPU (HIM interface)
- Interrupt Controller

Connections:
- tile_in
- tile_out
- lldd_in
- gdc_out
- of_in
- Frame IF
- Frame IF
- cfg
Core dataflow

Diagram:

- Grid Formatter
- DRAM
- Shadow
- LG Grid Calc
- Affine
- LG Grid Calc
- LLDD Calculator
- LLDD

Nodes:
- lg_out
- lghead
- (u,v), EOG...
- ag_out
- (u,v), EOG...
- gg_out
- (u,v), EOG...
- llld_out
- shd
- EOG
SGDU – Details 1

Distortions

- **Global distortions**: used to correct any lens imperfections and bring image to perfect geometry
- **Local distortions**: user defined, used for real-time image manipulation

Distortion Formatter

- Combines global and local distortions into a unified model and store it into local SRAM memory
SGDU – Details 2

GD Core(s)
- Coordinate mapping and image re-sampling
- Applies colour correction and generates alpha channel

Output Formatter(s)
- Write output image into external DRAM
- Applies alpha-blending if programmed
SGDU – Preliminary Specs

Input & Output

- 2 input video stream @ 60 fps: FullHD +20%
- Max input resolution @ 30 fps: 8M Pixels
- Max output resolution @ 120 fps: FullHD + 20%
- Up to 80MP resolution in still image mode.
- Support for 3 colour component / pixel, 8 and 16 bit/colour

Features:

- 1/32 sub-pixel computation precision
- 100% distortion correction: any input mapped to any output
- Independent distortion per colour plane
- Independent lens geometry correction and user defined distortions
- Exposure correction per colour plane & alpha channel support
# SGDU – Implementation Details

## Use Case Scenario 1
- **Input**: 3M Pixel 2048 x 1536
- **Output**: Full HD 1920 x 1080
- **Frame Rate**: 30 fps
- **Colour**: YUV 4:2:0, 8 bit
- **GDE**: 3 GDC cores parallel

<table>
<thead>
<tr>
<th>GDE HW Size</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic</td>
<td>450 k Gates</td>
</tr>
<tr>
<td>Internal RAM</td>
<td>400 k Bits</td>
</tr>
<tr>
<td>System</td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td>7.83 MB</td>
</tr>
<tr>
<td>Bandwidth*</td>
<td>263 MB/s</td>
</tr>
<tr>
<td>Latency**</td>
<td>T ms</td>
</tr>
</tbody>
</table>

*Memory bandwidth includes: reading input image, writing output image and overheads

## Use Case Scenario 2
- **Input**: 3M Pixel 2048 x 1536
- **Output**: Full HD 1920 x 1080
- **Frame Rate**: 30 fps
- **Colour**: YUV 4:2:0, 8 bit
- **GDE**: 1 GDC core sequential

<table>
<thead>
<tr>
<th>GDE HW Size</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic</td>
<td>200 k Gates</td>
</tr>
<tr>
<td>Internal RAM</td>
<td>160 k Bits</td>
</tr>
<tr>
<td>System</td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td>7.83 MB</td>
</tr>
<tr>
<td>Bandwidth*</td>
<td>263 MB/s</td>
</tr>
<tr>
<td>Latency**</td>
<td>3T ms</td>
</tr>
</tbody>
</table>

**Latency to be determined**
Applications – Wide Field of View Camera
Electronic Zoom Pan & Tilt - eZPT

eZPT Engine
P = 30 deg,
Z = 1x,
T = 0 deg

eZPT Engine
P = 0 deg
Z = 3x,
T = 0 deg

eZPT Engine
P = -20 deg,
Z = 2x,
T = -30 deg
WFoV and Face Based Cropping in Conferencing

eZPT Engine

WFoV and Face Based Cropping in Conferencing

eZPT Engine

WFoV and Face Based Cropping in Conferencing

eZPT Engine
Engineers at Play (1)
Engineers at Play (2)
Longer Demo