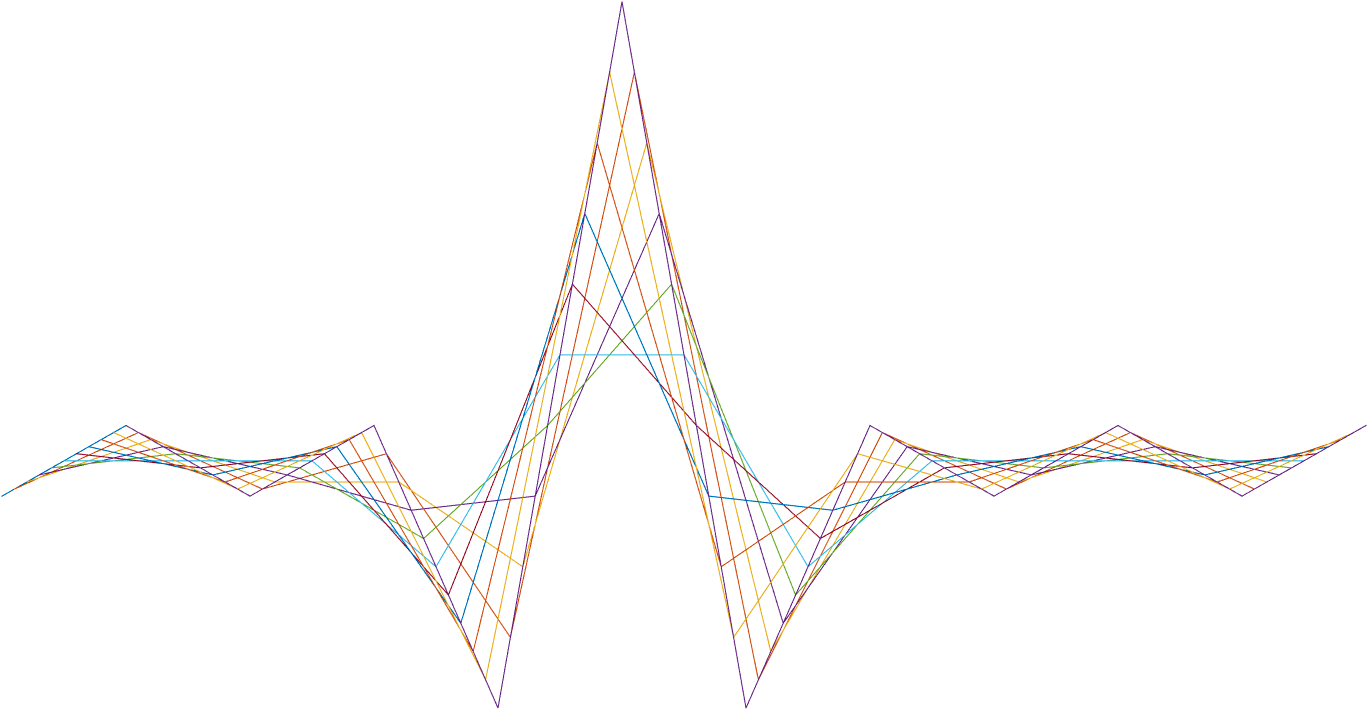
IVCAM2.0 3D Imaging Camera



ASIC A0 JFIL Geometric filter specification

27 September 2016

Revision 0.5.0

Intel Top Secret

Table 1: Revision history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Matlab Version | Revision Number | Revised by | Description | Revision Date |
| 2016a | 0.5.0 | David Silver | Initial release | July 4, 2016 |
|  | 0.5.1 | Ohad Menashe | Logic, registers, 64 templates | 21-08-2016 |
|  | 0.5.1 | Ohad Menashe | Logic, registers | 25-08-2016 |
| 0.7.38 | 0.7 | Ohad Menashe | Logic refinement | 22-09-2016 |
| 0.6.37 | 0.7 | Yoni Chechik | RegsJFILgeomBadConfVal -> RegsJFILgeomBadConf | 10.11.16 |

7

Contents

[1. Introduction 3](#_Toc455659982)

[1.1 General 3](#_Toc455659983)

[2. Interfaces 4](#_Toc455659984)

[1.2 Memory interfaces 4](#_Toc455659985)

[1.3 Other block interfaces 4](#_Toc455659986)

[1.4 Input 4](#_Toc455659987)

[1.4.1 Output 4](#_Toc455659988)

[2 Detailed Description 5](#_Toc455659989)

[2.1 Outline 5](#_Toc455659990)

[2.2 Windows structure 5](#_Toc455659991)

[2.3 Template structure 5](#_Toc455659992)

[2.4 Template matching computation 7](#_Toc455659993)

[2.5 Averaging over the neighborhood 7](#_Toc455659994)

[2.6 Pipe flow of a single pixel 8](#_Toc455659995)

[3. Registers 9](#_Toc455659996)

[2.7 New Registers definition 9](#_Toc455659997)

[3 Memory and computation 10](#_Toc455659998)

[3.1 Memory requirement 10](#_Toc455659999)

[3.2 Computational load 10](#_Toc455660000)

Tables

[Table 1: Revision history 1](#_Toc455660006)

[Table 2: Registers 9](#_Toc455660007)

1. Introduction

General

The Geometric Filter is located inside the *Depth Filter* block, before the *bilateral filter (DQ).* It determines which pixels should be considered valid and which invalid based on the morphological structure of the neighborhood.

1. Interfaces

Memory interfaces

The block require memory interface for 5 line buffers

Other block interfaces

Input

The block receive the following as input from the RUV block:

* 3x3 patch – 16b depth
* 12b ir (pass-through)
* 3x3 patch – 4b confidence
* flags – 4bit
  + Usage: TBD

### Output

The block send output to the bilateral filter in the Depth Filter block:

* 16b depth
* 12b ir (pass-through)
* 4b Confidence
* flags – 4bit
  + Usage: TBD

# Memory

## Memory requirement

As this filter uses a 5x5 patch for depth, IR and confidence, a two column buffer needs to store previous data for each of the three inputs. The size of each column is defined in RegsGNRLimgVsize

# Detailed Description

## Outline

The geometric filter operates on a depth frame + a binary validity mask created from the confidence and the preset value RegsJFILgeomConfThr using the following calculation:

binaryMask = confidencePatch>RegsJFILgeomConfThr

The binary mask is generated from pixels either below register-set confidence level.

The filter uses a 5x5 sliding window to determine the geometry of a pixel’s neighborhood binary mask. The geometry is decided base on bit-wise correlation to pre-defined templates of different shapes.

For each pixel the best template is matched based on bit wise correlation and the central pixel depth and confidence values are calculated based on the neighboring pixels.

We note the pixel validity as passing/failing to pass the preset threshold of RegsJFILgeomConfThr. As such, there are two distinct cases that needs to be differentiated:

* Pixel should be invalidated - the template central pixel is false.  
  Since throughout the pipe all the pixels are valid, an invalidation process is in fact setting the confidence to a preset low value RegsJFILgeomBadConf.
* Pixel should be validated – the template central pixel is true. Since the pixel can be already valid, the confidence is set as the median of the neighbors (which can either raise or lower the value).

In a case the selected template preform invalidation, the pixel confidence is updated under these conditions:

* The pixel confidence is lower RegsJFILgeomGoodConf
* The pixel confidence is higher than RegsJFILgeomBadConf
* The match score is higher than RegsJFILgeomMinHits

If all the above are true, we assign it the confidence value from the preset register RegsJFILgeomBadConf.

In a case the selected template preform validation, the pixel depth and confidence are updated under these conditions:

* The pixel confidence is lower RegsJFILgeomGoodConf
* The match score is higher than RegsJFILgeomMinHits
* The pixel’s selected template is enabled through the 64 bit-value RegsJFILgeomTemplateEnable
  + In case the selected template is disabled, the next highest template is selected

If all the above are true, we assign it the confidence and depth values using a median of the pixel’s neighbors.  
The pixel’s neighbors are selected using the *template median mask*, which is coupled with the selected template.  
Note that the maximum neighbors for each pixel is set to maximum 8 neighbors (self+8).

## Windows structure

We take 5x5 sliding window, totaling in 25 depth values per pixel.

## Template structure

We use up to 64 binary templates, of various edge directions examples illustrated in **Figure 1-2a**. The set of templates can be updated by RAM. The templates are coming in pairs: for each template there is



Figure 1-2a



Figure 1-2b

## Template matching computation

We take for each pixel its neighborhood of 5x5 validity mask that was based on the confidence value and confidence threshold.

Then we calculate the score function between every pixel to each of the templates. This is done as follows - we define the score function  between two binary vectors to be:

Then we calculate for each pixel, the vector  of score functions between that pixel 5x5 neighborhood and each template:



If the template that has the highest score from that pixel neighborhood, has a score above a predefined threshold, we mark it as matched to that template.

If two templates reach the same score, the template with the lower index should be taken. In case the template is masked by the RegsJFILgeomTemplateEnable, the score for the template is not calculated and is set to 0.

## Averaging over the neighborhood

If a neighborhood is matching a template, we take the values of that neighborhood according to the *template median mask* acting as an unmasking filter (the unmask pixels are the dark red in **Figure 1-2b**. Illustrates in Figure 1-3

Figure 1-3



1. Registers

| **Name** | **Size** | **Default** | **Range** | **Special values/ description** |
| --- | --- | --- | --- | --- |
| **GNRL** |  |  |  |  |
| RegsGNRLimgVsize | 10 | 480 | [120-960] |  |
| **JFIL** |  |  |  |  |
| RegsJFILgeomBypass | 1 | 0 | {0,1} |  |
| RegsJFILgeomMinHits | 5 | 17 | [0-25] | Minimum score indicating valid template match |
| regsJFILgeomBadConf | 4 | 3 | [0-15] | Confidence value to assign to invalid pixels |
| regsJFILgeomGoodConf | 4 | 3 | [0-15] | Confidence threshold for untouched pixels |
| regsJFILgeomTemplateEnable\_00 | 32 | 0xffffffff | [0-0xffffffff] | 64 bits indicating which templates are enabled when updating a valid depth pixel |
| regsJFILgeomTemplateEnable\_01 |
| RegsJFILgeomConfThr | 4 | 7 | [0-15] | Confidence value indicating whether a pixel is valid or not |

Table 2: Registers

1. Test Plan

| **Name** | **Range** | **Distribution** |
| --- | --- | --- |
| **JFIL** |  |  |
| RegsJFILgeomBypass | 0 | **1%** |
| 1 | **99%** |
| RegsJFILgeomMinHits | 0-7 | 10% |
| 8-15 | 20% |
| 15-20 | 50% |
| 20-25 | 20% |
| regsJFILgeomBadConf | 0-3 | 40% |
| 4-9 | 40% |
| 10-15 | 20% |
| regsJFILgeomGoodConf | 0-3 | 50% |
| 4-9 | 30% |
| 10-15 | 20% |
| regsJFILgeomTemplateEnable\_00 | [0-0xffffffff] | 100% |
| regsJFILgeomTemplateEnable\_01 | [0-0xffffffff] | 100% |
| RegsJFILgeomConfThr | 0-3 | 25% |
| 4-7 | 25% |
| 8-11 | 25% |
| 12-15 | 25% |

Table 2: Randomization