An FPGA-based pipeline for micropolarizer array imaging

Pierre-Jean Lapray, Luc Gendre, Alban Foulonneau, Laurent Bigue

An FPGA-based pipeline for Micro-Polarizer Array imaging

Pierre-Jean Lapray, Luc Gendre, Alban Foulonneau, and Laurent Bigué

IRIMAS institute, 12 rue des frères lumière, 68093 Cedex Mulhouse, France

Abstract

The enhancement of current camera performances, in terms of framerate, image resolution and pixel width, has direct consequences on the amount of resources needed to process video data. Stokes imaging permits to estimate polarization of light and create multiple polarization descriptors of the scene. Therefore, such video cameras need fast processing for critical applications like overseeing, defect detection or surface characterization. An FPGA hardware implementation of Stokes processing is presented here that embeds dedicated pipeline for micropolarizer array sensors. An optimized fixed-point pipeline is used to compute polarimetric images, i.e. Stokes vector, degree of polarization and angle of polarization. Simulation and experimental studies are done. The hardware design contains parallel processing, low latency and low power and could meet actual real-time and embeddable requirements for smart camera systems.

Keywords: Stokes imaging, micro-polarizer array, hardware implementation, FPGA

1. Introduction

Analyzing the polarization of the light coming directly from a source or scattered by an object, using an efficient polarimeter instrument, has become of great interest. Due to their nature, polarimeters provide information that are not available with conventional imaging systems. It is used for example in astrophysics [1, 2, 3], remote sensing [4], interferometry [5], biomedical applications [6, 7, 8], or nanostructures and metamaterials characterization [9, 10]. Their benefits are growing bigger as the technology allows faster, more detailed, and more precise measurements [11].

Polarization of light is linked to the wave-propagation vector of the electromagnetic waves. Stokes theory [12] is a method for describing polarization properties of light. In this formalism, the polarization is totally described by a four-components vector, called Stokes vector and commonly denoted $S = [s_0\ s_1\ s_2\ s_3]^T$.

Stokes imaging is done by using one imaging sensor (or several sensors, depending on the technology) and several optical elements, like linear polarizers, wave plates or retarders,

*Corresponding author

Email address: pierre-jean.lapray@uha.fr (Pierre-Jean Lapray, Luc Gendre, Alban Foulonneau, and Laurent Bigué)

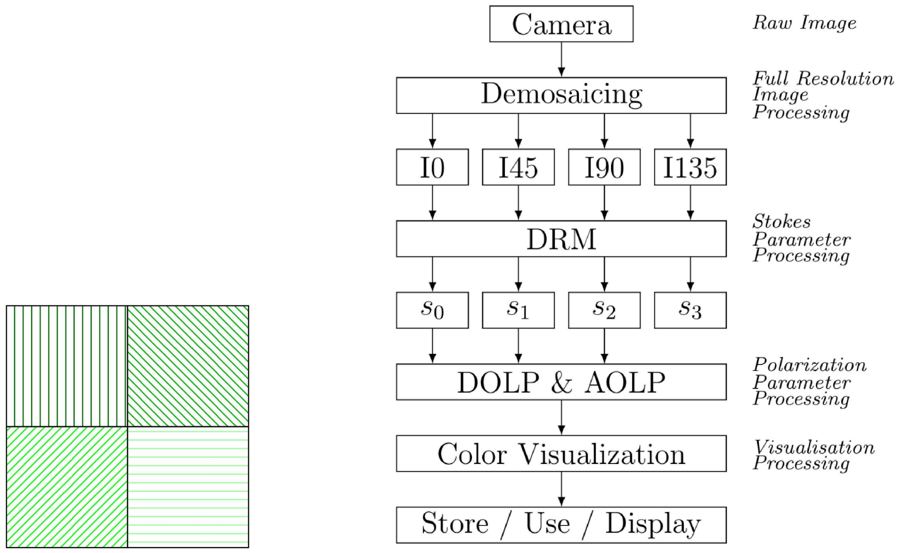
prisms, liquid crystals, etc. Each pixel of the imaging system needs to be processed in order to bring out, finally, the four components of the Stokes vector. Linear polarimeter is the class of device that is designed to measure only the first three polarization Stokes parameters: s_0, s_1, and s_2. These parameters are stored in full resolution images, and are used to calculate other useful descriptors like degree of linear polarization ($DOLP$) or angles of linear polarization ($AOLP$).

There are different imaging device architectures that allow the polarization to be analyzed, each of which has its own drawbacks and advantages. A review of recent acquisition systems for polarimetric imaging is done in Table 1. The same diversity of instruments exists for multispectral acquisition systems [13]. There are two main methods to acquire multi-channel polarimetric images: the scanning technique and the snapshot technique. The scanning technique implies that multiple polarimetric information are acquired successively in time. Snapshot could give multiple polarization states at the same time and allows for video acquisition and direct processing/visualization. Nowadays, the snapshot imaging instruments have become more and more exploited, especially with the Micro-Polarizer Array (MPA) device (e.g. the PolarCam from 4D technology [14]), due to its compactness. Polarimetric imaging using MPA recently gains in maturity to become out-of-the-lab instruments.

Table 1: Summary of the acquisition methods for passive Stokes imaging.

Method	Recent Work	Full	Compact
Scan (division-of-time)			
Rotatable Retarder & Fixed Polarizer (RRFP)	[15]	[✓]	[✗]
One Liquid-Crystal Variable Retarder & fixed linear polarizer (LCVR)	[16, 17, 18, 19]	[✓]	[✓]
Two Liquid-Crystal Variable Retarders & fixed linear polarizer (LCVRs)	[20, 21, 22, 23]	[✓]	[Depend]
Liquid-crystal variable retarder	[17, 18, 19]	[✓]	[✓]
Acousto-Optic Tunable Filter (AOTF)	[24]	[✓]	[✗]
Snapshot			
Division-of-Amplitude (DoAmP)	[25]	[✓]	[✗]
Division-of-Aperture (DoAP)	[26]	[✓]	[✓]
Division-of-Focal-Plane and Micro-Polarizer Array (DoFP & MPA)	[27, 28, 29, 30, 31]	[Depend]	[✓]
Canonical Refraction (CR) / Biaxial Crystal (BC)	[32, 33]	[✓]	[✗]
Channeled Imaging Polarimeters (CIP)	[34, 35]	[✓]	[✗]

The industry is demanding more and more requirements about efficient image processing, low-power and low-cost camera architecture. On this, we can add the emergence of embedded systems dedicated to applications such as video protection, medical imaging or driving assistance. This gives operators the ability to make decision faster. Regarding the enhancements for 20 years in terms of image sensor resolution (e.g. actual 8K format), framerate or dynamic range, the snapshot technique seems adapted but could contain relatively high throughput of data to process. To reduce the volume of data to be transmitted by restricting only the information that the user deems relevant, some cameras have the possibility to do image processing in real-time. We deduce that there is a need to have an efficient



(a) (b)

Figure 1: (a) The super pixel spatial arrangement of the MPA considered in this work. The pattern is uniformly repeated over all of the photosensitive cells. (b) Global architecture pipeline. It includes four processing steps.

polarimetric imaging pipeline, as it was done for other imaging techniques in the past few decades, e.g. [36, 37]. We have not found complete and comprehensive works dealing with Stokes imaging on FPGA; here is the subject of this article.

The paper is organized as follows; in Section 2, we start by proposing a Stokes imaging pipeline dedicated to MPA, that will be embedded in a smart camera. Then, we present the hardware design of the pipeline in Section 3. Finally, we analyze the efficiency of the solution by a complete implementation of the pipeline in an FPGA in Section 4 before concluding in Section 5.

2. Stokes imaging pipeline

The MPA design that we are considering in the present paper corresponds to the pattern presented in Figure 1(a). It is composed of pixel size linear polarizers oriented at 0°, 45°, 90°, 135°, superimposed on a camera sensor chip. Therefore, each pixel measures only one of the four different intensities, called polarization states, depending on the orientation of the polarizer in front of the considered pixel. The polarization states are named hereafter I_0, I_{45}, I_{90}, I_{135}. With this setup, a single image acquisition gives a mosaiced image providing partial spatial information on each of the polarization states simultaneously. A few computation steps are needed to estimate the incoming polarization at full picture resolution from such an image. We propose here a pipeline dedicated to MPA. Although we consider a precise MPA architecture, the whole pipeline can still be applied on other MPA architectures with the only change of the data reduction matrix (**DRM**) described below, such as MPA that would allow the circular polarization component to be estimated in the future. This pipeline will then be adapted in an efficient hardware design in Section 3 using VHDL (VHSIC Hardware Description Language). The pipeline is summarized as a block diagram on Figure 1(b), which is composed of the following elements:

- A demosaicing block, composed of an interpolation method to retrieve the full spatial resolution of the intensity data,
- A reduction matrix processing, that outputs the Stokes vector parameters in parallel,
- *DOLP* (Degree Of Linear Polarization) and *AOLP* (Angle Of Linear Polarization) modules for recovering polarimetric descriptors,
- A visualization processing block that outputs useful qualitative information, taking into account the human visual system.

Stokes imaging is based on irradiance measurements. So it intrinsically includes all issues that arise from the standard imaging radiometry domain. If we do not correct for fixed pattern noise (i.e. dark noise and photo response non-uniformity), similar noise consequences as conventional radiometric imaging could occur. But some recent sensors often have embedded noise corrections within the chip to prevent these effects. Additionally, if no proper polarimetric calibration is done for the data reduction matrix, variations on transmission and extinction ratio of the polarimetric elements are not taken into account. Thus the polarization descriptors could be miscalculated. Complete calibration of micro-polarizer array cameras can be found in the literature [38], along with the impact of noise in polarimetric applications [39]. In the whole pipeline, we assume that images from the MPA camera are calibrated and do not need pre-processing (i.e. radiometric calibration, linearization, dark correction, flat-field, etc.).

2.1. Estimation from measurements

In the current paper, the Stokes vector $\mathbf{S}$ is used to represent the polarization of the light [12]. There are other possible representations [40] that will not be discussed here.

$$\mathbf{S} = [s_0 \;\; s_1 \;\; s_2 \;\; s_3]^T \tag{1}$$

with s_0 the total light intensity, s_1 the intensity difference through a 0° and 90° polarizers, s_2 the intensity difference through a 45° and −45° polarizers, and s_3 referring to left or right handedness of the polarized light.

When the light is coming from a source or a surface to a polarimeter, the vector $\mathbf{I}$ that represents measured intensities by the sensor can be described as follows:

$$\mathbf{I} = \mathbf{M}.\mathbf{S} \tag{2}$$

where $\mathbf{M}$ is the measurement matrix, defined during system calibration. A Data Reduction Matrix (DRM) [41] can be defined for reconstruction of the input signal $\mathbf{S}$ such as:

$$\hat{\mathbf{S}} = \mathbf{DRM}.\mathbf{I} \qquad \text{with} \;\; \mathbf{DRM} = \mathbf{M}^{+} \tag{3}$$

where $\mathbf{M}^+$ is the pseudo-inverse of the measurement matrix.

Using Eq. (3), the Stokes vector can be recovered from a set of at least four intensities. Using only linear polarizers in the optical setup will not allow the s_3 component to be estimated [42][1]. We are precisely in that case with the polarimeter system we are considering in this paper, since the MPA is composed of only linear polarizers. Even though the system provides four different polarization states, only the three first Stokes vector elements s_0, s_1, s_2 can be computed. For the rest of the paper, we will only consider polarization descriptors that can be computed from these three elements.

2.2. Descriptor computation

From the Stokes vector parameters s_0, s_1, s_2, the following quantities can be computed, that help understanding the nature of the polarization.

The Degree Of Linear Polarization ($DOLP$) represents the amount of linear polarization in the light beam. It takes values between zero for non polarized light and one for totally polarized light, intermediate values referring to partial polarization.

$$DOLP = \frac{\sqrt{s_1^2 + s_2^2}}{s_0} \tag{4}$$

The azimuthal angle of linear polarization ($AOLP$) is also computed from the Stokes vector. It represents the angular orientation of the main axis of the polarization with respect to the chosen angular reference used for system calibration :

$$AOLP = \frac{1}{2} \arctan\left(\frac{s_2}{s_1}\right) \tag{5}$$

2.3. Visualization application

An interesting application that could be done when performing Stokes imaging is the color visualization of data. It is an application in the sense that the visualization is a direct interpretation of light polarization by the user. It is well known that some insects and animals can have the polarization vision capacities. Bio-inspired techniques to map the polarization signature into a color representation has been widely studied [43, 44]. In this work, we implemented the Tyo *et al.* method [45], that is probably the most common method from the state-of-art. It is based on the HSV (Hue, Saturation, Value) color data fusion that map polarization features to the HSV space as follows:

$$AOLP \rightarrow H \qquad DOLP \rightarrow S \qquad s_0 \rightarrow V \tag{6}$$

Hue is associated with the angle of polarization; the connection between hue and AOLP is the circularity behavior of data. Example of this mapping will be shown in the next section. The main drawback is that a pixel could sense light properties with both low irradiance and high polarization state, but this specificity can't really be represented along this technique, because s_0 is mapped to the image pixel intensity. It is corrected in a recent work [46].

[1] In most imaging applications, the circular polarization magnitude is very low.

2.4. Demosaicing

In case of a snapshot camera using MPA with a mosaiced pattern of filters [14], each pixel has a different instantaneous field of view (IFOV)[2]. In other words, a single pixel only senses a fraction of the total polarization states, so the other missing polarization states have to be interpolated. If we compute Stokes parameters without using a spatial interpolation method among channels, it causes severe artifacts such as zipping or aliasing (especially when viewing $DOLP$), and makes computer vision algorithms to fail. Due to the regularity of an MPA filter pattern, it is easy to define convolution kernels applied to each polarization channel separately. It is well known that bilinear interpolation could avoid a lot of IFOV problems [47]. Moreover, this is known to be efficient and computationally simple, and thus could be implemented in real-time. More evolved demosaicing algorithms that are designed for Color Filter Array (CFA) could not be used directly, because polarimetric imaging does not have significant correlation among channels when capturing a randomly polarized scene. We propose to evaluate five kernels and build a choice for the final implementation.

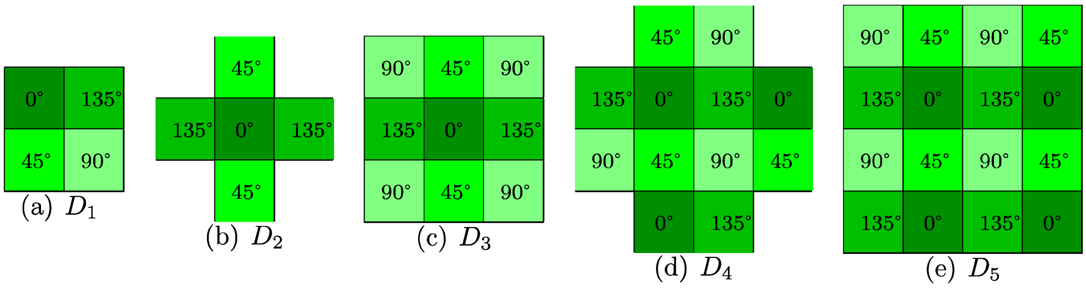


Figure 2: Visualization of the five demosaicing kernels D_{1-5} used across the evaluation. It refers to the neighborhood used for interpolation. Each pixel records only $I0$, $I45$, $I90$ or $I135$ light polarization states.

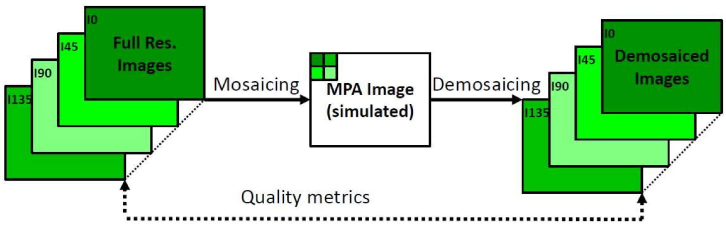
2.4.1. Demosaicing method evaluation

Here, we are interested in evaluating the five demosaicing kernels and their influence on the resulting image quality. These methods are described in a recent work by Ratliff *et al.* [47]. Kernels can be visualized in Figure 2. In this past evaluation study [47], only IFOV artifacts were measured using purely simulated data, and modulation/intermodulation transfer function as evaluation metrics. To select which methods we should use for any application, we made an evaluation using more quality metrics. We argue that a more comprehensive assessment using a larger number of metrics is missing, and that the use of objective and subjective metrics is useful for selecting a demosaicing algorithm. Indeed, the key of our evaluation is to use well-known and benchmarked metrics that have been already used for CFA imaging [48], excepted for perceptual color difference metrics, that is not applicable in our case. We propose to use these four indicators: PSNR (peak signal to noise ratio), SSIM [49] (Structural SIMilarity), RMSE (root mean squared error) and

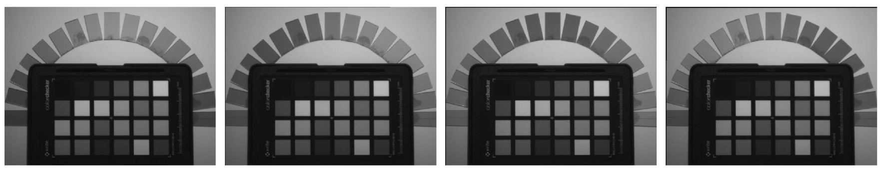
[2] This step could be by-passed in case of having a polarimeter with already full resolution polarization images at its output (using a division-of-aperture polarimeter for example).

correlation [50] metrics. PSNR has a clear physical meaning and is commonly used in computer science for compression and reconstruction evaluation in digital image processing. Higher score means better image quality. SSIM has a better perceptual matching, where best image quality is achieved by a score near to one. It is typically a modified MSE metric where errors are penalized according to their visibility in the image. Perceptual quality is not straightforward to measure at all, but to our knowledge SSIM tends to be a well benchmarked method. RMSE defines the square root of average square deviation between the original and reconstructed image. The cross-correlation criterion (between 0 and 1) gives similar quality results independently if an offset exists among intensities, where better score means higher reconstruction quality. These metrics are fully described in [48].

According to the application target, some of these metrics could be preferred to select proper algorithm independently for its signal to noise ratio, its structural similarity or its better correlation results.



(a) Evaluation pipeline



(b) I0 (c) I45 (d) I90 (e) I135

Figure 3: (a) Pipeline for the evaluation of interpolation kernels. (b), (c), (d), (e) Full resolution images used for the demosaicing evaluation. Images were captured using a gray-level sensor and linear polarization filter. The scene is composed of a hand-made polarization chart with pieces of linear polarizers arranged in half circle (polarization axis in the lengthiness of the pieces), and a X-Rite Passport color checker (with patches that are relatively highly diffuse, thus unpolarized).

About the methodology: Figure 3(a) presents the pipeline used for evaluation. A set of images acquired with a gray-level camera was first taken. A linear polarizer in front of the camera is rotated to 0°, 45°, 90° and 135° using a motion controlled instrument (the Agilis™Conex-AG-PR100P piezo rotation mount from Newport). The resolution of images is 1024×768 pixels. A tungsten lamp is used for the illuminant. It is assumed that placing a filter in front of a camera in different positions could cause optical image translation. The four images are registered using a simple correlation-based registration from the state-of-art [51].

An MPA image could be represented by a mosaiced image with sampled polarization component. One polarization state is sensed by spatial pixel location. For the simulation, the four full resolution images are combined to simulate an MPA image. The spatial arrangement selected is that of the commercial MPA camera from 4D technology [14]. When mosaiced image is generated, we apply the five demosaicing kernels D_1 to D_5. So, we recover 5×4 spatially interpolated images corresponding to the five kernels for each of the four polarization states. After that, images are compared with the full resolution images (ground truth) by applying the selected metrics. To be more consistent, we also apply these verification to all parameters and descriptor images described in Section 2, namely on s_0, s_1, s_2, $DOLP$, $AOLP$ and HSV (Hue Saturation Value) visualization of polarization.

2.4.2. Demosaicing method analysis

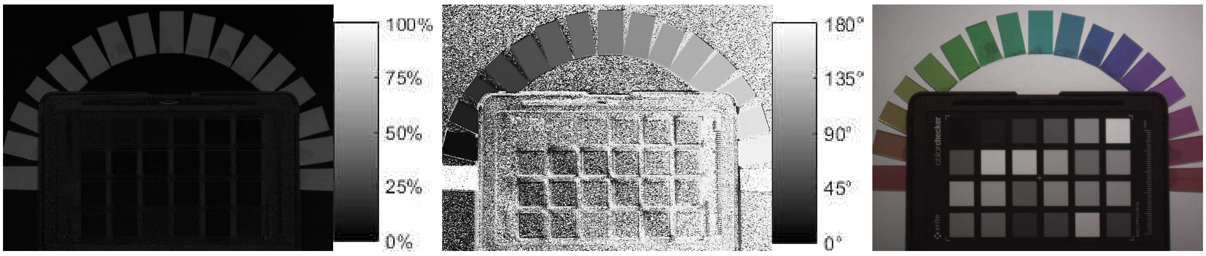
Visualization of the results are summed up in Figure 4. For an exhaustive visualization of the results, all image resulting from all methods are shown in the appendix in Figure A.1. By looking at the reconstructed intensity image s_0 in Figure A.1(a), we can see that D_4 and D_5 images look blurry, whereas D_{1-3} preserve edges. It could be simply explained by the fact that the kernels used are larger (4×4 pixels), and that pixel values are estimated using largest neighborhood. The HSV color visualization in Figures 4(n) to 4(r) is also interesting because we can see by zooming that all methods feature some color artifacts and chromatic aberrations that could also appear in CFA images. About D_2, and by looking at the cross at the center of the color checker, we can distinguish a lot of zipper effects [52].

By looking more particularly at the $DOLP$ images in Figures 4(i) to 4(m), we see that the zipper effect is very pronounced for kernel D_1 and D_2 and is the least marked for kernel D_4 and D_5. Hence we verify the fact that D_4 gives the best results concerning the removing of IFOV artifacts according to [47], even in $AOLP$. Kernel D_5 is not giving the best results because it intrinsically contains a symmetric structure in the kernel (see Figure 2), whereas D_4 breaks this symmetry by removing the corner pixel factors in the filter processing.

The quantitative evaluation results are presented in Table 2. We find that all $AOLP$ images have very bad scores. This is due to the fact that the arc tangent operation is a circular operation, which can lead to very different values in the case where an angle is calculated in the part of the image where $DOLP$ is very small (see Figures 4(d) and 4(e)). Globally, the different metrics seem to be correlated; all the results clearly show that D_3 is the best interpolation method for most images tested and most metrics. Thus we selected it to be implemented in our design.

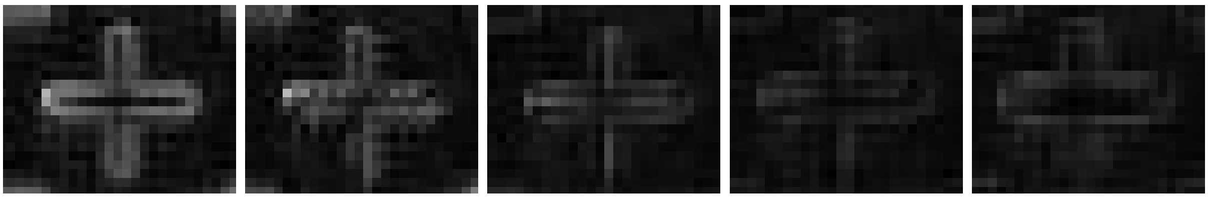
In applications such as computer vision (e.g. semantic segmentation, image dehazing, image denoising, etc.), it is important to preserve perfect edge information, thus we will prefer the method which gives less artifacts. Moreover, applications with natural scenes containing a lot of moving objects would prefer to use D_4, because the effects of IFOV artifacts are often more pronounced in these conditions. In other applications that need accurate measurements like in machine vision or computer graphics (metallic object defect detection, diffuse/specular separation, rendering, etc.), we would prefer D_3.

(a) Full resolution s_0 (b) Full resolution s_1 (c) Full resolution s_2



(d) Full resolution $DOLP$ (e) Full resolution $AOLP$ (f) Full resolution HSV

(g) Full resolution zoomed $DOLP$ (h) Full resolution zoomed HSV



(i) D_1 (j) D_2 (k) D_3 (l) D_4 (m) D_5

(n) D_1 (o) D_2 (p) D_3 (q) D_4 (r) D_5

Figure 4: (a-f) Full resolution images used as reference for the demosaicing evaluation. (i-m) Zoomed $DOLP$ demosaicing results. (n-r) Zoomed HSV demosaicing results. Demosaicing is done using the five kernels applied on the test images (shown in Figure 3). The zoomed region corresponds to the white cross at the center of the color checker. We can see zipper effect and different magnitude of IFOV artifacts due to demosaicing method. The full resolution images are shown on Figure A.1.

PSNR	D_1	D_2	D_3	D_4	D_5
I0	35.7	37.9	42.1	37.6	37.1
I45	36.1	38.5	44.0	37.5	36.6
I90	35.5	37.9	43.0	37.0	36.1
I135	35.9	38.3	43.7	38.1	37.5
S0	38.6	40.8	45.2	38.0	37.6
S1	38.6	41.0	45.9	45.9	43.3
S2	39.0	41.5	47.2	46.9	43.9
DOLP	25.6	28.1	33.1	33.6	31.0
AOLP	7.0	7.2	7.3	6.6	6.3
HSVvis	30.4	32.6	36.1	34.0	33.5

SSIM	D_1	D_2	D_3	D_4	D_5
I0	0.96	0.97	0.98	0.97	0.97
I45	0.97	0.98	0.99	0.97	0.97
I90	0.96	0.97	0.99	0.97	0.97
I135	0.96	0.98	0.99	0.98	0.97
S0	0.98	0.98	0.99	0.98	0.97
S1	0.93	0.95	0.98	0.98	0.97
S2	0.93	0.96	0.98	0.99	0.97
DOLP	0.72	0.76	0.84	0.84	0.80
AOLP	0.28	0.30	0.34	0.26	0.23
HSVvis	0.92	0.94	0.97	0.95	0.95

RMSE	D_1	D_2	D_3	D_4	D_5
I0	0.016	0.013	0.008	0.013	0.014
I45	0.016	0.012	0.006	0.013	0.015
I90	0.017	0.013	0.007	0.014	0.016
I135	0.016	0.012	0.007	0.012	0.013
S0	0.012	0.009	0.005	0.013	0.013
S1	0.012	0.009	0.005	0.005	0.007
S2	0.011	0.008	0.004	0.005	0.006
DOLP	0.052	0.039	0.022	0.021	0.028
AOLP	0.447	0.436	0.432	0.466	0.482
HSVvis	0.030	0.024	0.016	0.020	0.021

Corr.	D_1	D_2	D_3	D_4	D_5
I0	1.00	1.00	1.00	1.00	1.00
I45	0.99	1.00	1.00	1.00	1.00
I90	0.99	1.00	1.00	1.00	1.00
I135	1.00	1.00	1.00	1.00	1.00
S0	1.00	1.00	1.00	1.00	1.00
S1	0.87	0.90	0.94	0.96	0.95
S2	0.86	0.90	0.94	0.96	0.94
DOLP	0.63	0.76	0.88	0.90	0.89
AOLP	0.39	0.43	0.47	0.41	0.38
HSVvis	0.99	0.99	0.99	0.99	0.99

Table 2: Demosaicing results for kernels D_{1-5} and the four metrics. Best scores are highlighted in green whereas bad scores in red.

3. Hardware design

3.1. Global architecture

Here we describe the complete hardware architecture that composes our system. It is derived from the pipeline from the previous section, which is shown on Figure 1(b).

3.1.1. Demosaicing

The demosaicing process requires a pixel with the intensities of its neighborhood to estimate the missing intensities. The filtering which is described in VHDL is shown on Figure 5. This work is developed for our particular MPA images containing polarizers arranged as shown on Figure 1(a). It could be extended and adapted to any other MPA filter design (without loss of generality).

We use the 3×3 filtering mask $\mathbf{F}$ described below and sampled channel images $\mathbf{P_k}(\mathbf{I_{raw}}(i))$, where i indexes the 1-D pixel position in the raw image $\mathbf{I_{raw}}$, and k indexes the angles of polarization $\{0°, 45°, 90°, 135°\}$. We define the sampling function $\mathbf{P_k}$, where locations of available channels in a mosaiced image $\mathbf{I_{raw}}$ are sampled as:

$$\mathbf{P_k}(\mathbf{I_{raw}}(i)) = \begin{cases} \mathbf{I_{raw}}(i) \text{ if channel } k \text{ is at pixel position } i \text{ in } \mathbf{I_{raw}} \\ 0 \text{ otherwise.} \end{cases} \tag{7}$$

where $k \in \{0°, 45°, 90°, 135°\}$.

Now let us consider the convolution filter [48]:

$$\mathbf{F} = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{8}$$

We can now compute each channel component $\hat{\mathbf{I}}_\mathbf{k}$ using the same convolution filter $\mathbf{F}$, along with the sampled image plane $\mathbf{P_k}$ as this:

$$\hat{\mathbf{I}}_\mathbf{k} = \mathbf{F} * \mathbf{P_k}(\mathbf{I_{raw}}), \tag{9}$$

For the hardware design, we need two FIFO buffers to store the first two image rows, and six shift registers that are responsible for holding the eight neighboring pixels for the current pixel interpolation. The serial connection of the FIFO memories emulates the vertical displacement of the mask. The transfer of values from the FIFO to the shift registers emulates the horizontal scrolling. The nine pixels are multiplied by their corresponding coefficients in $\mathbf{F}$ using nine products. Then, eight accumulators add those pixels. Shift registers perform single clock delay in order to respect the pipeline timing coherency across pixels. The output streaming pixels for the corresponding $\mathbf{F} \times \mathbf{P_k}(i)$ is finally transmitted to the rest of the pipeline.

The bilinear filtering processing is applied four times in the hardware design, as we have to interpolate spatial data for recovering the four polarization images $\hat{\mathbf{I}}_\mathbf{k}$. The four masks $\mathbf{P_k}$ are created directly from the input pixel stream $\mathbf{I_{raw}}(i)$, by multiplexing the channel intensities. We take one pixel out of two and one line out of two and let other pixels to zero. It is important to note that this design could be easily adapted to other demosaicing methods, by changing the $\mathbf{F}$ coefficients, and extending or reducing the neighborhood.

3.1.2. DRM

Figure 6 shows the VHDL entity of the DRM module. This module is responsible for the Stokes parameter computation s_{0-3}, as described in Section 2. Inputs are global common signals (*pixel_clk* and *reset*) and pixel stream $\hat{\mathbf{I}}_\mathbf{k}$ from the demosaicing block.

In case of using a sensor that provides directly I_0, I_{45}, I_{90} and I_{135}, a simplified DRM could be used, as this:

$$\mathbf{DRM} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

For other sensors that do not provide directly these specific polarization angles, or when polarizing elements are not considered to be ideals, a calibration step must be done to recover the proper DRM matrix [53] prior to measurements.

3.2. Stokes parameters

Stokes processing needs the data to be manipulated with decimal numbers. From there, there are several possibilities. We will have to take into account the precision required for

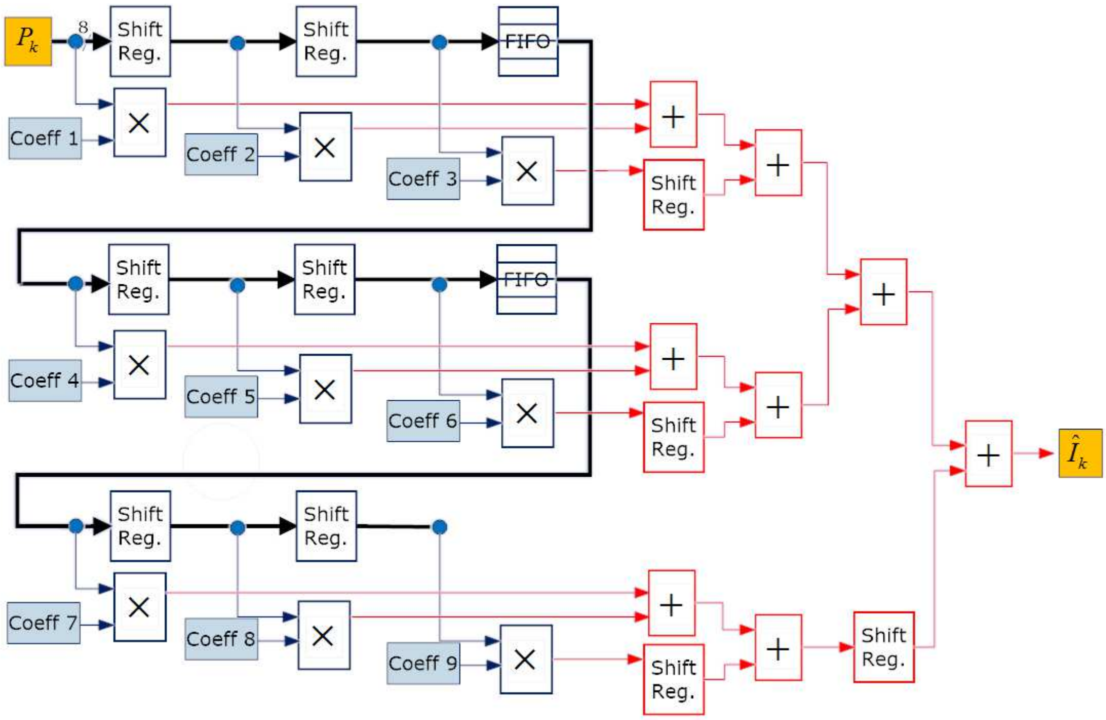


Figure 5: Demosaicing block used in our experiment. It proceeds with a 3×3 window of neighboring pixels. Coefficients are from those of Equation 8 in our hardware implementation.

our calculations, to know approximately the range of values that will be used. Fixed-point and floating-point formats could be considered. The representation of decimal numbers in the CPU and GPU architecture is underlying and all numbers and manipulation of numbers are done using single or double precision representations with the IEEE 754 floating-point standard. We are aware that some new FPGA architectures are coming on the market by embedding hardware blocks dedicated to floating point computation (e.g. Arria 10 from Altera). Nevertheless, these devices are very expensive and are still in a niche market. For a common FPGA architecture, the designer can choose his own mode of representation. Maximizing the accuracy along with the bit-depth is an optimization procedure, resulting in low complexity, low power and increasing the maximum operating frequency of the system.

AOLP and *DOLP* image processing have been described using the IEEE fixed-point library included in the VHDL 2008 standard. The computation of these components requires resource consuming and time consuming operators, like divisions (computationally expensive in hardware real-time design), an arc tangent and a square root computation. For the division operator, it could not be bypassed, so we use the divider contained in the VHDL fixed-point library. For the square root and arc tangent implementations, there are three possible methods :

1. using CORDIC (COordinate Rotation DIgital Computer) algorithm [54],
2. using a polynomial approximation,
3. using a customizable LUT.

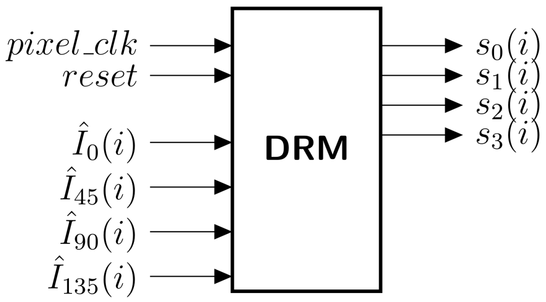


Figure 6: Entity of the Data Reduction Matrix block (DRM). It is the first block dedicated to Stokes processing.

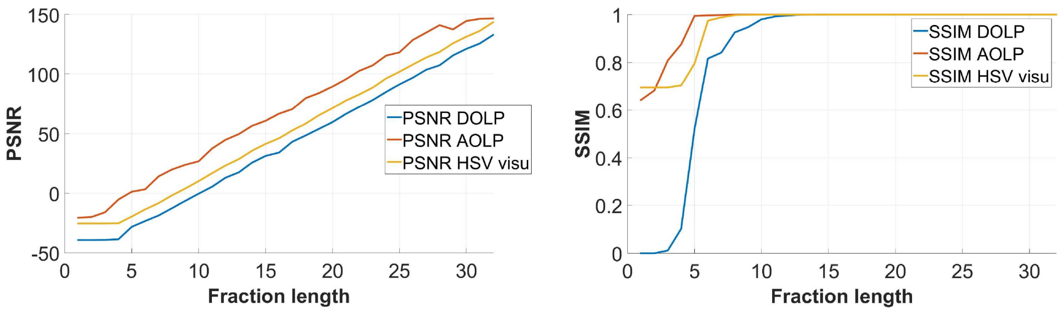
The CORDIC algorithm is known to be the most hardware efficient method for the implementation of trigonometric, hyperbolic and square root equations [55]. It only needs shift-add handling, which is the less time/resource consuming. It avoids additional multipliers and dividers, which are widely used for a polynomial approximation. Cordic is directly available in FPGA software design tools on the market. The problem could be the big latency introduced; typically it is 32 clock cycles in our system. With a $125MHz$ clock, it corresponds to $0.26\mu s$ which is very low but could be significant in hard constrained applications.

If the user wants a very low latency system, a LUT implementation with a one clock cycle per operation would be preferred. This technique consumes a lot of LUT blocks to support the possible input dynamic range of values (e.g. $s_1^2 + s_2^2$ for the square root), and needs bigger FPGA with sufficient LUT resources. In the rest of our work, we choose the Cordic algorithm, as we want to keep the maximum precision, along with low hardware resource utilization, and avoid dividers for the system.

3.3. Fixed-point study

A study on how to select the appropriate bit-depth at the expense of image quality is done. PSNR and SSIM quality metrics are applied on images resulting directly from fixed-point operations, i.e. $DOLP$, $AOLP$ and HSV images (see Section 2 for description). As s_0, s_1 and s_2 are integer, it is easy to define the pixel bit-depth required before the radix point. s_1 and s_2 are varying between -255 and $+255$, whereas s_0 is varying between 0 and 510. We know that $DOLP$ is varying between 0 and 1, so we deduce that the $s_1^2 + s_2^2$ operation should not have dynamic greater than 260 100. That means that 18 *bits* are necessary for the integer part to keep the best accuracy.

From that point, we could evaluate PSNR and SSIM for the other processed images using an increasing number of bit after the radix point. Native Matlab fixed-point numeric objects are constructed and used through the whole processing pipeline. We varied the length of the decimal part of the numbers, incrementing by 1, starting from an accuracy of 0-bit for the fraction length, and going up to 32-bit precision. All results are then compared with the floating-point processing using cast as double type in Matlab. Metrics are then applied between the fixed-point generated images and double-type processing images. The results of these comparisons are shown in Figure 7. With this method, we could select proper accuracy of our calculations, depending on the word length and fraction length. For our pipeline and for the rest of the paper, we selected 14 bits as fractional depth. It is assumed that typical PSNR values for an 8-bit image and with a relatively good quality, range between 20 and $40dB$ [56].



(a) PSNR (b) SSIM

Figure 7: Fixed-point Matlab study results on polarimetric descriptors, by extending the bit-depth of the fixed-point fractional part.

3.4. Hardware simulation

After describing the pipeline in hardware, simulation is done. The method is based on co-simulation using Simulink HDL Verifier conjointly with Modelsim Vsim (VHDL simulator) from Mentor®. The simulation environment in Simulink is shown on Figure 8. The mosaiced image data, the same as in Section 2.4, is sent to the simulator in a streaming manner. Image data is first arranged as 1-D vector using the frame-to-packet Simulink block. Then, an unbuffer serializes data at the rate of one pixel per clock tick. The whole VHDL design is interpreted inside Modelsim and the processed output is hence sending back to Simulink and all output images are displayed/saved.

4. Experimental results

In this section, the design is now implemented on an FPGA board and tested with a video from an MPA sensor.

4.1. Implementation

Results of the complete implementation of the pipeline design is presented in Table 3. We implemented the design targeting the Zedboard (xc7z020 Zynq-7000 FPGA) with Xilinx® Vivado tool. This FPGA has a total of $85K$ programmable logic cells, $4.9Mb$ of block RAM and 220 DSP Slices.

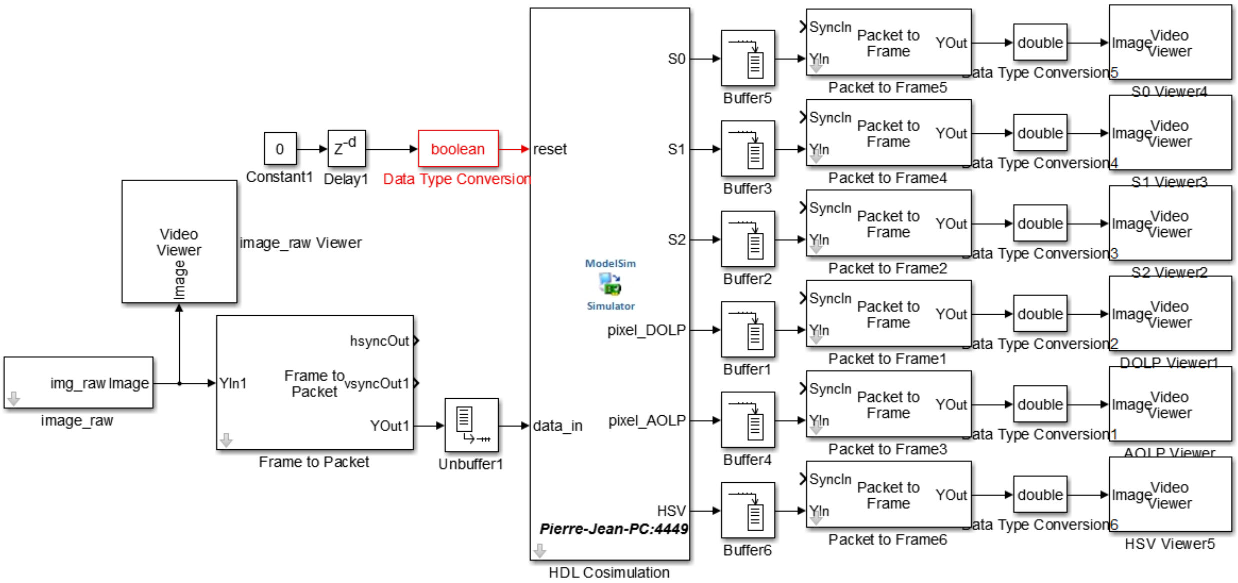


Figure 8: Simulation environment used to simulate the complete pipeline design in Figure 1(b).

4.2. Experimental setup

Video sample used for the experiment was taken from the PolarCam by 4D technology [14]. The full resolution is 648×488 and 8-bit per pixel. We assume that the camera output is linear and that there is no need to produce additional dark and flat corrections for using the data. The captured scene is composed of pieces of linear polarizers stuck on a glass, that are moved by hand in front of the camera.

To verify the hardware implementation, Simulink was used along with the FPGA-in-the-loop (FIL) tool. The FIL tool is a communication interface that sends the streaming video data to the FPGA via JTAG connection (approximately $13Mbit/s$ of transferring bandwidth), and the FPGA sends it back to the CPU after processing. As the FPGA processes the data faster (125MHz) than the JTAG bandwidth, it contains a *clock enable* which is synchronized and activated/deactivated depending on the load of the JTAG data buffer (responsible for transmitting the data). The processed data is then retrieved in the FIL tool and saved/displayed into Matlab workspace.

The video results, showing the outputs of our hardware pipeline, are available online [3].

4.3. Discussion

Summaries of hardware implementation reports of our design are shown in Table 3 and 4. It appears that *DOLP* and *AOLP* are blocks that consume the most of resources. This is due to the implementation of CORDIC for the square root and arc tangent. The demosaicing process consumes 956 slice LUTs for four filtering operations. We compared our utilization report with the one that would be implemented using a C++-based synthesized

[3] http://pierrejean.lapray.free.fr/MPA_HW_polarimetry/

Table 3: Detailed report of hardware implementation of the imaging pipeline on the Zynq xc7z020clg484-1.

	Used	Available	Percentage
Power Consumption	0.55 W		
Logic utilization			
Number of Occupied Slices	3,211	13,300	24.1%
Complexity distribution			
Number of Slice registers	7,328	106,400	6.9%
Number of Slice LUTs	9,558	53,200	18.0%
Demosaicing	956		1.8%
DRM processing	225		0.4%
DOLP processing	4,311		8.1%
AOLP processing	4,066		7.6%
Number of DSP	12	220	5.5%
Number of FIFO/BRAMBs	4	140	2.9%
Number of DCM-ADVs	1	4	25%

Table 4: Summary of hardware implementation reports on several Xilinx devices for comparison.

FPGA	Artix-7 (xc7a200t)		Kintex-7 (xc7k325t)		Virtex-7 (xc7vx690t)		Zynq (xc7z045)	
Power consumption (W)	0.50		0.51		0.68		0.58	
	Number	Utilization	Number	percentage	Number	Utilization	Number	Utilization
Slices	3,211	9.5%	3,149	6.2%	3,141	2.9%	3,204	5.9%

design, i.e. the High Level Synthesis (HLS) tool from Xilinx. We found that four bilinear filters implemented targeting the same FPGA chip consume 1817 slice LUTs, which is more compared to our implementation (956 slice LUTs). This is due to the inherent complexity added (bus and buffer structure around the processing block) by HLS when the design is synthesized.

In terms of performance, pixel latencies are variable depending on blocks. For the demosaicing block, the latency is two times the image width plus three, because pixel can not be computed since enough neighboring pixels are available in buffers. Other processing latencies are low as each processing block is pipelined. Fixed point limited precision permits to perform one operation per clock cycle, even for dividers. Respectively, it takes 4, 40 and 39 clock cycles to process DRM, $DOLP$ and $AOLP$. The color visualization is not time consuming as it is just a combination of s_0, DOLP and AOLP outputs. The total pixel latency needed is 1343 clock cycles for the 648×488 resolution, that corresponds to $10.74\mu s$ at $125MHz$ in our case. This latency could meet a lot of fast response needs in machine vision and industry applications.

Table 5: Comparison among the existing state-of-the-art works.

Work	Architecture	Power consumption	Frame processing time	Output
[57]	GPU (GeForce 9400 GS)	$\approx 50W$	$33.6ms$	S_0, S_1, S_2, DOLP
[58]	8-core DSP	$18W$	$17.0ms$	S_0, S_1, S_2, AOLP, DOLP, HSV
[59]	FPGA	$2.45W$	$20.0ms$	S_0, AOLP, DOLP
Ours	FPGA	$0.55W$	$16.6ms$	S_0, S_1, S_2, AOLP, DOLP, HSV

All designs tested in Table 3 can process the pixel stream using a maximum frequency of 125MHz (this was the required frequency during place and route steps) without introducing timing problems, i.e. no negative setup or hold slacks in the paths. So any combination of image resolution and framerate that could match this maximum streaming pixel clock constraint is achievable. For example, a 1080p format with a resolution of 1920×1080 at 60 frames per second can be considered, as it needs $1920 \times 1080 \times 60 = 124\,416\,000$ operations per second to process the streams. We want to point out that due to blank video timing, processing pixel clock can be different and thus lower than the video pixel clock that is usually specified in the standard video timing requirements.

Table 5 shows the comparison among different state-of-the-art realizations of efficient Stokes imaging processing. It appears that our work can achieve better performance with minimal power consumption compared to other state-of-the-art works.

5. Conclusion

We proposed the design of a Stokes imaging pipeline in FPGA dedicated to MPA. We validated the processing blocks in hardware simulation using Simulink/Modelsim, and made studies about fast interpolation methods and fixed-point approximations. We tested the pipeline in real conditions using a Zynq implementation, and showed different implementation resource utilization among existing Xilinx FPGAs. The hardware-dedicated pipeline is capable of processing all Stokes vectors plus numerous already analyzed polarimetric descriptors at an achievable 1080p60 format, and a low fixed latency. The design has a low hardware complexity, low latency, and the achievable performance is promising for future high performance embedded cameras and critical applications.

As future work, the design will be interfaced with a camera communication protocol, using the standard interface GigeVision, a framebuffer and a simple streaming interface (AXI stream or Avalon-stream) bus. Many standard interfaces as Gigevision are not directly available, and have to be purchased or developed. A straightforward solution would be to use the system-on-chip FPGA capability of Zynq, which embeds a processor architecture (a Dual-core ARM Cortex-A9 MPCore) and logic blocks. A Linux driver for interfacing the GigeVision protocol along with a memory bridge that share data from user-space Linux memory to the FPGA side would be a solution.

References

[1] T. Gehrels, et al., Planets, stars and nebulae: studied with photopolarimetry, Vol. 23, University of Arizona Press, 1974.

[2] K. Nagendra, J. Stenflo, Solar polarization, Vol. 243, Springer Science & Business Media, 2013.

[3] L. Kolokolova, J. Hough, A.-C. Levasseur-Regourd, Polarimetry of stars and planetary systems, Cambridge University Press, 2015.

[4] J. S. Tyo, D. L. Goldstein, D. B. Chenault, J. A. Shaw, Review of passive imaging polarimetry for remote sensing applications, Applied optics 45 (22) (2006) 5453–5469.

[5] M. Novak, J. Millerd, N. Brock, M. North-Morris, J. Hayes, J. Wyant, Analysis of a micropolarizer array-based simultaneous phase-shifting interferometer, Appl. Opt. 44 (32) (2005) 6861–6868. `doi:10.1364/AO.44.006861`.
URL `http://ao.osa.org/abstract.cfm?URI=ao-44-32-6861`

[6] N. Ghosh, I. A. Vitkin, Tissue polarimetry: concepts, challenges, applications, and outlook, Journal of biomedical optics 16 (11) (2011) 110801–11080129.

[7] T. Novikova, A. Pierangelo, A. De Martino, A. Benali, P. Validire, Polarimetric imaging for cancer diagnosis and staging, Optics and photonics news 23 (10) (2012) 26.

[8] M. Dubreuil, P. Babilotte, L. Martin, D. Sevrain, S. Rivet, Y. Le Grand, G. Le Brun, B. Turlin, B. Le Jeune, Mueller matrix polarimetry for improved liver fibrosis diagnosis, Optics letters 37 (6) (2012) 1061–1063.

[9] T. W. Oates, T. Shaykhutdinov, T. Wagner, A. Furchner, K. Hinrichs, Mid-infrared gyrotropy in split-ring resonators measured by mueller matrix ellipsometry, Optical Materials Express 4 (12) (2014) 2646–2655.

[10] D. Schmidt, Characterization of highly anisotropic three-dimensionally nanostructured surfaces, Thin Solid Films 571 (2014) 364–370.

[11] M. D. Presnar, J. P. Kerekes, Modeling and measurement of optical polarimetric image phenomenology in a complex urban environment, in: Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International, IEEE, 2010, pp. 4389–4392.

[12] G. Stokes, On the composition and resolution of streams of polarized light from different sources, Transactions of Cambridge Philosophical Society 9 (1852) 339–416.

[13] P.-J. Lapray, X. Wang, J.-B. Thomas, P. Gouton, Multispectral filter arrays: Recent advances and practical implementation, Sensors 14 (11) (2014) 21626–21659. `doi:10.3390/s141121626`.
URL `http://www.mdpi.com/1424-8220/14/11/21626`

[14] N. J. Brock, C. Crandall, J. E. Millerd, Snap-shot imaging polarimeter: performance and applications, Vol. 9099, 2014, pp. 909903–909903–12. `doi:10.1117/12.2053917`.
URL `http://dx.doi.org/10.1117/12.2053917`

[15] D. H. Goldstein, Mueller matrix dual-rotating retarder polarimeter, Appl. Opt. 31 (31) (1992) 6676–6683. `doi:10.1364/AO.31.006676`.
URL `http://ao.osa.org/abstract.cfm?URI=ao-31-31-6676`

[16] A. Peinado, A. Lizana, J. Campos, Optimization and tolerance analysis of a polarimeter with ferroelectric liquid crystals, Appl. Opt. 52 (23) (2013) 5748–5757. `doi:10.1364/AO.52.005748`.
URL `http://ao.osa.org/abstract.cfm?URI=ao-52-23-5748`

[17] L. Gendre, A. Foulonneau, L. Bigué, Full stokes polarimetric imaging using a single ferroelectric liquid crystal device, Optical Engineering 50 (2011) 50 – 50 – 10. `doi:10.1117/1.3570665`.
URL `http://dx.doi.org/10.1117/1.3570665`

[18] W. A. Woźniak, M. Pretka, P. Kurzynowski, Imaging stokes polarimeter based on a single liquid crystal variable retarder, Appl. Opt. 54 (20) (2015) 6177–6181. `doi:10.1364/AO.54.006177`.
URL `http://ao.osa.org/abstract.cfm?URI=ao-54-20-6177`

[19] J. M. López-Téllez, N. C. Bruce, O. G. Rodríguez-Herrera, Characterization of optical polarization properties for liquid crystal-based retarders, Appl. Opt. 55 (22) (2016) 6025–6033. `doi:10.1364/AO.55.006025`.
URL `http://ao.osa.org/abstract.cfm?URI=ao-55-22-6025`

[20] F. Goudail, P. Terrier, Y. Takakura, L. Bigué, F. Galland, V. DeVlaminck, Target detection with a liquid-crystal-based passive stokes polarimeter, Appl. Opt. 43 (2) (2004) 274–282. doi:10.1364/AO.43.000274.
URL http://ao.osa.org/abstract.cfm?URI=ao-43-2-274

[21] J. M. Bueno, Polarimetry using liquid-crystal variable retarders: theory and calibration, Journal of Optics A: Pure and Applied Optics 2 (3) (2000) 216.

[22] O. Aharon, I. Abdulhalim, Liquid crystal wavelength-independent continuous polarization rotator, Optical Engineering 49 (3) (2010) 034002–034002–4. doi:10.1117/1.3366545.
URL http://dx.doi.org/10.1117/1.3366545

[23] M. Vedel, S. Breugnot, N. Lechocinski, Full stokes polarization imaging camera, in: Proc. SPIE, Vol. 8160, 2011, p. 81600X.

[24] N. Gupta, D. R. Suhre, Acousto-optic tunable filter imaging spectrometer with full stokes polarimetric capability, Appl. Opt. 46 (14) (2007) 2632–2637. doi:10.1364/AO.46.002632.
URL http://ao.osa.org/abstract.cfm?URI=ao-46-14-2632

[25] E. Compain, B. Drevillon, Broadband division-of-amplitude polarimeter based on uncoated prisms, Appl. Opt. 37 (25) (1998) 5938–5944. doi:10.1364/AO.37.005938.
URL http://ao.osa.org/abstract.cfm?URI=ao-37-25-5938

[26] T. Mu, C. Zhang, Q. Li, R. Liang, Error analysis of single-snapshot full-stokes division-of-aperture imaging polarimeters, Opt. Express 23 (8) (2015) 10822–10835. doi:10.1364/OE.23.010822.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-23-8-10822

[27] J. S. Tyo, Hybrid division of aperture/division of a focal-plane polarimeter for real-time polarization imagery without an instantaneous field-of-view error, Optics letters 31 (20) (2006) 2984–2986.

[28] G. Myhre, W.-L. Hsu, A. Peinado, C. LaCasse, N. Brock, R. A. Chipman, S. Pau, Liquid crystal polymer full-stokes division of focal plane polarimeter, Opt. Express 20 (25) (2012) 27393–27409. doi:10.1364/OE.20.027393.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-20-25-27393

[29] K. A. Bachman, J. J. Peltzer, P. D. Flammer, T. E. Furtak, R. T. Collins, R. E. Hollingsworth, Spiral plasmonic nanoantennas as circular polarization transmission filters, Opt. Express 20 (2) (2012) 1308–1319. doi:10.1364/OE.20.001308.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-20-2-1308

[30] X. Zhao, A. Bermak, F. Boussaid, V. G. Chigrinov, Liquid-crystal micropolarimeter array for full stokes polarization imaging in visible spectrum, Opt. Express 18 (17) (2010) 17776–17787. doi:10.1364/OE.18.017776.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-18-17-17776

[31] W.-L. Hsu, G. Myhre, K. Balakrishnan, N. Brock, M. Ibn-Elhaj, S. Pau, Full-stokes imaging polarimeter using an array of elliptical polarizer, Opt. Express 22 (3) (2014) 3063–3074. doi:10.1364/OE.22.003063.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-22-3-3063

[32] A. Peinado, A. Lizana, A. Turpín, C. Iemmi, T. K. Kalkandjiev, J. Mompart, J. Campos, Optimization, tolerance analysis and implementation of a stokes polarimeter based on the conical refraction phenomenon, Opt. Express 23 (5) (2015) 5636–5652. doi:10.1364/OE.23.005636.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-23-5-5636

[33] I. Estévez, V. Sopo, A. Lizana, A. Turpin, J. Campos, Complete snapshot stokes polarimeter based on a single biaxial crystal, Opt. Lett. 41 (19) (2016) 4566–4569. doi:10.1364/OL.41.004566.
URL http://ol.osa.org/abstract.cfm?URI=ol-41-19-4566

[34] K. Oka, T. Kato, Spectroscopic polarimetry with a channeled spectrum, Opt. Lett. 24 (21) (1999) 1475–1477. doi:10.1364/OL.24.001475.
URL http://ol.osa.org/abstract.cfm?URI=ol-24-21-1475

[35] K. Oka, T. Kaneko, Compact complete imaging polarimeter using birefringent wedge prisms, Opt. Express 11 (13) (2003) 1510–1519. doi:10.1364/OE.11.001510.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-11-13-1510

[36] P. J. Lapray, B. Heyrman, M. Rossé, D. Ginhac, Hdr-artist: High dynamic range advanced real-time imaging system, in: 2012 IEEE International Symposium on Circuits and Systems, 2012, pp. 1428–1431. doi:10.1109/ISCAS.2012.6271513.
[37] P.-J. Lapray, B. Heyrman, D. Ginhac, Hdr-artist: an adaptive real-time smart camera for high dynamic range imaging, Journal of Real-Time Image Processing 12 (4) (2016) 747–762. doi:10.1007/s11554-013-0393-7.
URL http://dx.doi.org/10.1007/s11554-013-0393-7
[38] S. B. Powell, V. Gruev, Calibration methods for division-of-focal-plane polarimeters, Opt. Express 21 (18) (2013) 21039–21055. doi:10.1364/OE.21.021039.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-21-18-21039
[39] F. Goudail, M. Boffety, Fundamental limits of target detection performance in passive polarization imaging, J. Opt. Soc. Am. A 34 (4) (2017) 506–512. doi:10.1364/JOSAA.34.000506.
URL http://josaa.osa.org/abstract.cfm?URI=josaa-34-4-506
[40] D. Goldstein, Polarized Light, 2nd Edition, Marcel Dekker, New York & Basel, 2003.
[41] M. Bass, E. W. Van Stryland, D. R. Williams, W. L. Wolfe, Handbook of optics, Vol. 2, McGraw-Hill New York, 2001.
[42] J. S. Tyo, Optimum linear combination strategy for an n-channel polarization-sensitive imaging or vision system, J. Opt. Soc. Am. A 15 (2) (1998) 359–366. doi:10.1364/JOSAA.15.000359.
URL http://josaa.osa.org/abstract.cfm?URI=josaa-15-2-359
[43] G. D. Bernard, R. Wehner, Functional similarities between polarization vision and color vision, Vision Research 17 (9) (1977) 1019 – 1028. doi:http://dx.doi.org/10.1016/0042-6989(77)90005-0.
URL http://www.sciencedirect.com/science/article/pii/0042698977900050
[44] L. B. Wolff, Polarization camera for computer vision with a beam splitter, J. Opt. Soc. Am. A 11 (11) (1994) 2935–2945. doi:10.1364/JOSAA.11.002935.
URL http://josaa.osa.org/abstract.cfm?URI=josaa-11-11-2935
[45] J. S. Tyo, E. N. Pugh, N. Engheta, Colorimetric representations for use with polarization-difference imaging of objects in scattering media, J. Opt. Soc. Am. A 15 (2) (1998) 367–374. doi:10.1364/JOSAA.15.000367.
URL http://josaa.osa.org/abstract.cfm?URI=josaa-15-2-367
[46] J. S. Tyo, B. M. Ratliff, A. S. Alenin, Adapting the hsv polarization-color mapping for regions with low irradiance and high polarization, Opt. Lett. 41 (20) (2016) 4759–4762. doi:10.1364/OL.41.004759.
URL http://ol.osa.org/abstract.cfm?URI=ol-41-20-4759
[47] B. M. Ratliff, C. F. LaCasse, J. S. Tyo, Interpolation strategies for reducing ifov artifacts in microgrid polarimeter imagery, Opt. Express 17 (11) (2009) 9112–9125. doi:10.1364/OE.17.009112.
URL http://www.opticsexpress.org/abstract.cfm?URI=oe-17-11-9112
[48] O. Losson, L. Macaire, Y. Yang, Comparison of color demosaicing methods, Advances in Imaging and Electron Physics 162 (2010) 173–265. doi:10.1016/S1076-5670(10)62005-8.
URL https://hal.archives-ouvertes.fr/hal-00683233
[49] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Transactions on Image Processing 13 (4) (2004) 600–612. doi:10.1109/TIP.2003.819861.
[50] D. Su, P. Willis, Demosaicing of color images using pixel level data-dependent triangulation, in: Proceedings of Theory and Practice of Computer Graphics, 2003., 2003, pp. 16–23. doi:10.1109/TPCG.2003.1206926.
[51] G. D. Evangelidis, E. Z. Psarakis, Parametric image alignment using enhanced correlation coefficient maximization, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (10) (2008) 1858–1865. doi:10.1109/TPAMI.2008.113.
[52] B. K. Gunturk, J. Glotzbach, Y. Altunbasak, R. W. Schafer, R. M. Mersereau, Demosaicking: color filter array interpolation, IEEE Signal Processing Magazine 22 (1) (2005) 44–54. doi:10.1109/MSP.2005.1407714.
[53] J. S. Tyo, D. L. Goldstein, D. B. Chenault, J. A. Shaw, Review of passive imaging polarimetry for

remote sensing applications, Appl. Opt. 45 (22) (2006) 5453–5469. doi:10.1364/AO.45.005453.
URL http://ao.osa.org/abstract.cfm?URI=ao-45-22-5453

[54] J. E. Volder, The cordic trigonometric computing technique, IRE Transactions on Electronic Computers EC-8 (3) (1959) 330–334. doi:10.1109/TEC.1959.5222693.

[55] R. Andraka, A survey of cordic algorithms for fpga based computers, in: Proceedings of the 1998 ACM/SIGDA Sixth International Symposium on Field Programmable Gate Arrays, FPGA '98, ACM, New York, NY, USA, 1998, pp. 191–200. doi:10.1145/275107.275139.
URL http://doi.acm.org/10.1145/275107.275139

[56] D. Salomon, Data compression: the complete reference, Springer Science & Business Media, 2004.

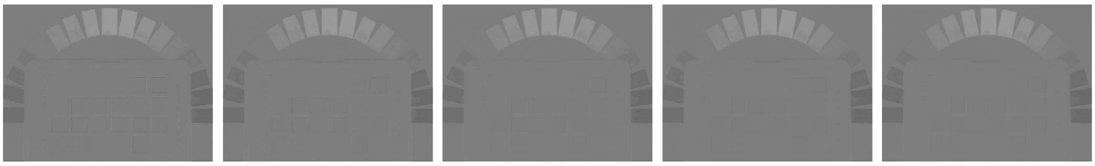
[57] H. Patel, Gpu accelerated real time polarimetric image processing through the use of cuda, in: Proceedings of the IEEE 2010 National Aerospace Electronics Conference, 2010, pp. 177–180. doi:10.1109/NAECON.2010.5712943.

[58] K. C.-K. Marcus Bednara, Real time polarization sensor image processing on an embedded fpga/multicore dsp system, Vol. 9506, 2015, pp. 9506 – 9506 – 13. doi:10.1117/12.2178823.
URL http://dx.doi.org/10.1117/12.2178823

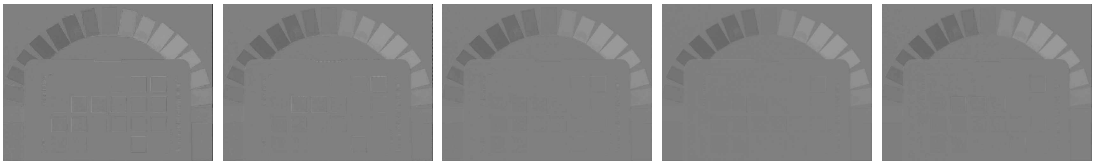
[59] V. G. Timothy York, Samuel Powell, A comparison of polarization image processing across different platforms, Vol. 8160, 2011, pp. 8160 – 8160 – 7. doi:10.1117/12.894633.
URL http://dx.doi.org/10.1117/12.894633

Appendix A. Demosaicing results

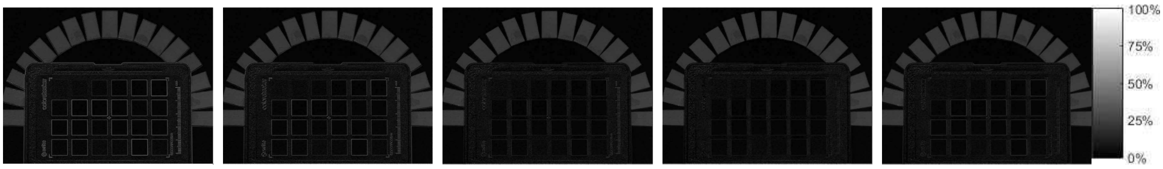
(a) s_0. The dynamic range $[0;511]$ is mapped to $[0;255]$ for visualization.



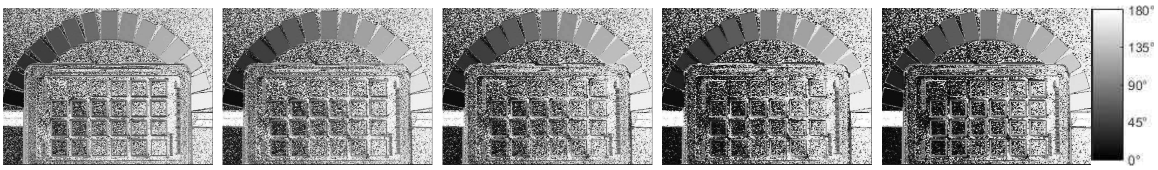
(b) s_1. The dynamic range $[-255;255]$ is mapped to $[0;255]$ for visualization.



(c) s_2. The dynamic range $[-255;255]$ is mapped to $[0;255]$ for visualization.



(d) DOLP



(e) AOLP

(f) HSV visualization

Figure A.1: Demosaicing results using the five kernels applied on the test images (shown in Figure 3). The five demosaicing methods D_{1-5} are described in Section 2.4. By zooming numerically on these images, we can see different magnitude of IFOVs artifacts due to demosaicing (especially for DOLP). (f) could be only visualized on the pdf color version of this paper.