Baumer
Passion for Sensors

Application Note

Calculation of polarization data using the Baumer GAPI SDK

AN201904/0.1/2019-02-26

Description

The Baumer polarizing cameras provide the ability to measure the intensity and the polarization of the acquired image. Therefore it enables you to utilize the polarization information in your application. This Application Note describes how to obtain the polarization information.

Products

Baumer cameras: VCXU-50MP, VCXG-50MP / Software: Baumer GAPI SDK from version v2.9.2

Contents

1 Technical Background

The Baumer polarization cameras are based on the Sony® IMC250MZR Sensor. This sensor is coated with a metal-mesh which filters the polarization information on 4 adjacent pixels. The polarization angle is filtered with an alignment of 0°, 45°, 90° and 135°.

With this information the following data can be calculated:

- Angle-Of-Polarization (AOP)
- Degree-Of-Linear-Polarization (DOLP)
- Angle-And-Degree-Of-Linear-Polarization (ADOLP)
- Image intensity

2 Usage of the polarizing camera with the Baumer GAPI SDK

The camera provides just the raw data about the polarization. The calculation of the different polarization formats is done on the host system with the Baumer GAPI SDK. This reduces the necessary bandwidth for the interface as the data is only transferred once and not for each polarization format (AOP, DOLP, ADOLP, intensity) separately.

2.1 Description of the polarization camera features

Using the standardizes SFNC Features “ComponentSelector” and “ComponentEnable” a GenICam™ compatible software can recognize that the camera provides raw polarized data. Those features cannot be changed (read-only).

For the identification the following features should be checked:

```
ComponentSelector = PolarizedRaw
ComponentEnable = True
```

With the aim to achieve compatibility to a wide range of 3rd party software, Baumer did not introduce custom image formats. The raw polarized data is transferred using the standard formats Mono8, Mono10, Mono12 or Mono12p.

Further, the camera offers features necessary for the calibration of the camera inside the category “Calibration Control”. Those features come filled with Baumer calibrated values. If necessary a “DeviceResetToDeliveryState” will reset the values to the calibrated values provided by Baumer.

GenICam™ Custom Baumer Features	
CalibrationControl	Category
CalibrationAngleOfPolarizationOffset (R/W)	The “CalibrationAngleOfPolarizationOffset” can be used to change the offset of the polarization angle to correct manufacturing tolerances as well as deviations in a camera system.
CalibrationMatrixValue (R/W)	The calibrated matrix to correct sensor effects and increase the accuracy of the polarization measurements. Usually the values should not be changed by the customer.
CalibrationMatrixValueSelector (R/W)	

Notice

The Features “CalibrationAngleOfPolarizationOffset” and “CalibrationMatrixValue” are written directly to the camera flash memory. This is compared to a normal feature access quite slow. Changing this values very regularly can also reduce the maximum lifetime of the flash memory.

2.2 Description of the Baumer GAPI SDK Features to calculate the polarization data

The calculation of the different polarization formats are done on the host system using the Baumer GAPI SDK.

The examples:

- 020_Polarized_SinglePart
- 021_Polarized_MultiPart

show step-by-step how this is done.

Notice

The calculation of the polarization formats is very demanding. To achieve a good performance the AVX2 vector extension of the processor is used. This is available from the Intel® Haswell and AMD Excavator architectures onwards. Without AVX2 the performance of the algorithm will be reduced drastically.

To ensure the calculation uses as little resources as possible and to achieve high frame rates there are two methods how to handle the data. Here we explain the most important configurations and the usage of the software features.

If only one of the polarization formats (AOP, DOLP or ADOLP) is needed for an application it is best to use a single-part image object. The single-part image object contains exactly one image.

```
// A Baumer Polarization Camera can be recognized by checking that the feature
// ComponentSelector has the value "PolarizedRaw"
if (pDevice->GetRemoteNodeList()->GetNodePresent("ComponentSelector"))
{
    if (pDevice->GetRemoteNode("ComponentSelector")->GetValue() == "PolarizedRaw")
    {
        // I'm a Polarization Camera
    }
}

// Acquire an image to a buffer
BGAPI2::Buffer* pBufferFilled = pDataStream->GetFilledBuffer(1000);
bo_uint width = static_cast<bo_uint>(pBufferFilled->GetWidth());
bo_uint height = static_cast<bo_uint>(pBufferFilled->GetHeight());
void* pBufferData = pBufferFilled->GetMemPtr();
bo_uint64 bufferDataSize = pBufferFilled->GetMemSize();
bo_uint64 imageOffset = pBufferFilled->GetImageOffset();
```

```
bo_uint64 imageDataSize = (bufferDataSize > imageOffset) ? (bufferDataSize - imageOffset) : 0;
void* pImageData = reinterpret_cast<char*>(pBufferData) + imageOffset;


// Create a new Image object using the data from the BGAPI buffer. The PixelFormat must
// be set to BaumerPolarized8/10/12 depending on the camera PixelFormat setting.
// e.g.: If the camera is set to the PixelFormat “Mono8”, the PixelFormat for the converted image
// must be set to “BaumerPolarized8”
pImageProcessor = new BGAPI2::ImageProcessor();
pImage = pImageProcessor->CreateImage(bufferWidth, bufferHeight, "BaumerPolarized8", pImageData,
imageDataSize);

// An existing image object can be reused which is more efficient than creating and
// destroying the object for each acquired image.
pImage->Init(bufferWidth, bufferHeight, "BaumerPolarized8", pImageData, imageDataSize);

// Select, which Polarized Format(AOP, DOLP, ADOLP or Intensity) you want to generate and disable
// all others.
BGAPI2::Node* pComponentSelector = pImage->GetNode("ComponentSelector");
BGAPI2::NodeMap*pComponents = pComponentSelector->GetEnumNodeList();
for (bo_uint64 i = 0; i < pComponents->GetNodeCount(); i++)
{
    pComponentSelector->SetInt(i);
    pImage->GetNode("ComponentEnable")->SetBool( pComponentSelector->GetValue() == "AOP" );
}

// Calculate the polarization format using a direct transformation from raw polarized
// image. For AOP and DOLP the converted Image must be Mono8, for ADOLP its RGB8
BGAPI2::Image* pComponent = pImageProcessor->CreateTransformedImage(pImage, "Mono8");
```

If more than one of the polarization formats (e.g. AOP und DOLP) is required in an application it is better to use a multi-part image object. This way all the calculations can be done in one go instead of calculating each format individually as shown above.

```
// Acquire an image to a buffer
BGAPI2::Buffer* pBufferFilled = pDataStream->GetFilledBuffer(1000);
bo_uint width = static_cast<bo_uint>(pBufferFilled->GetWidth());
bo_uint height = static_cast<bo_uint>(pBufferFilled->GetHeight());
void* pImageData = pBufferFilled->GetMemPtr();
bo_uint64 imageDataSize = pBufferFilled->GetMemSize();

// Create an Image object using the ImageProcessor
pImageProcessor = new BGAPI2::ImageProcessor();
pImage = pImageProcessor->CreateImage(bufferWidth, bufferHeight, "BaumerPolarized8", pImageData,
imageDataSize);
// or
pImage->Init(bufferWidth, bufferHeight, "BaumerPolarized8", pImageData, imageDataSize);
```

```
// Enable all polarized formats (AOP, DOLP, ADOLP, Intensity)
BGAPI2::Node* pCompSelector = pImage->GetNode("ComponentSelector");
BGAPI2::NodeMap*pComponents = pCompSelector->GetEnumNodeList();
for (bo_uint64 i = 0; i < pComponents->GetNodeCount(); i++)
{
    pCompSelector->SetInt(i);
    pImage->GetNode("ComponentEnable")->SetBool( true );
}

// Calculate all the polarization formats from the raw image to a multi-part Image object.
BGAPI2::Image* pMultiPartImage = pImageProcessor->CreateTransformedImage(pImage, "Mono8");

// Get necessary information about each multi-part
BGAPI2::Node* pComponentSelector = pMultiPartImage->GetNode("ComponentSelector");
BGAPI2::Node* pComponentEnable = pMultiPartImage->GetNode("ComponentEnable");
BGAPI2::Node* pComponentOffset = pMultiPartImage->GetNode("ComponentOffset");
BGAPI2::Node* pComponentLength = pMultiPartImage->GetNode("ComponentLength");
const void* const pImageBuffer = pMultiPartImage->GetBuffer();

// Loop through the multi-part Image object to extract all the parts
for (std::set<std::string>::const_iterator it = sComponents.begin(); it != sComponents.end();
it++)
{
    pComponentSelector->SetValue(it->c_str());

    std::string sComponent = pComponentSelector->GetValue().get();

    if (pComponentEnable->GetBool()) {
        bo_uint64 partLength = pComponentLength->GetInt();
        BGAPI2::Image* pComponent = NULL;
        if (partLength > 0)
        {
            // Part is valid
            bo_uint64 partOffset = pComponentOffset->GetInt();
            pComponent = pImageProcessor->CreateImage(width, height, "Mono8"
                , (char*)(pImageBuffer)+partOffset, partLength);
        }
        else
        {
            // Part is empty
            if (sComponent == "ADOLP") {
                // ADOLP is calculated from the AOP and DOLP images, therefore this
                // second transformation is necessary.
                pComponent = pImageProcessor->CreateTransformedImage(pMultiPartImage, "RGB8");
            }
        }
```

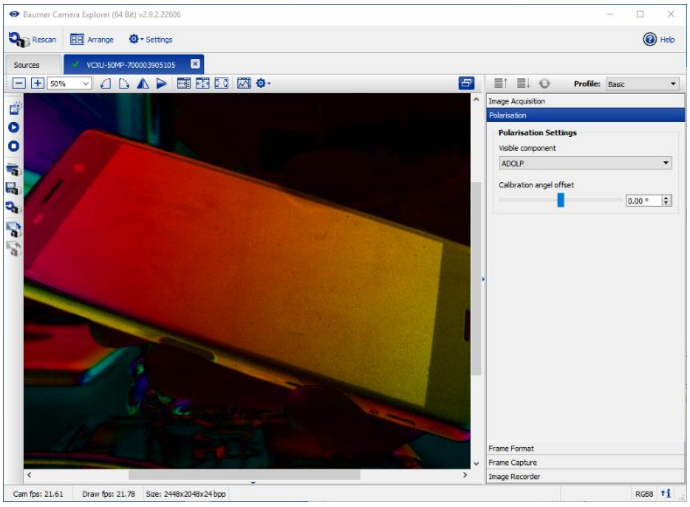
```
        if (pComponent)
        {
            // Whatever needs to be done with the polarization data goes here
            // doCustomCalculation(pComponent, sComponent);

            // Release the data when not needed anymore
            pComponent->Release();
        }
    }
}
```

3 Using the polarization camera with the Baumer Camera Explorer

The Camera Explorer can also be used to view and save polarization data in the formats AOP, DOLP, ADOLP and intensity.

The configuration is done in the Basic View, using the category Polarization.



4 Support

Please contact our Technical & Application Support Center with any questions.

Worldwide
Baumer Optronic GmbH
Badstrasse 30 · DE-01454 Radeberg
Deutschland

Phone +49 3528 4386 845
support.cameras@baumer.com

5 Legal Notes

Baumer Group

The Baumer Group is one of the worldwide leading manufacturers of sensors, encoders, measuring instruments and components for automated image processing. Baumer combines innovative technologies and customer-oriented service into intelligent solutions for factory and process automation and offers an unrivalled wide technology and product portfolio. With around 2,700 employees and 38 subsidiaries in 19 countries, the family-owned group of companies is always close to the customer. Baumer provides clients in most diverse industries with vital benefits and measurable added value by worldwide consistent high quality standards and outstanding innovative potential. Learn more at www.baumer.com on the internet.

Baumer
Passion for Sensors

Baumer Optronic GmbH
Badstrasse 30 · DE-01454 Radeberg
Phone +49 3528 4386 0 · Fax +49 3528 4386 86
sales@baumeroptronic.com · www.baumer.com

Document History

Date	Version	Name	Pages/ Chapter	Change
26.02.19	v0.1	pef, tosk, svo	all	creation of document, von lan unkontrolliert. Wenn noch was kommt, dann eben v0.2
10.04.19	v0.2	sba, lan	all	revision, S. 1 Aufzählung der Kameras vereinheitlicht, weitere kleine Änderungen