**PROJECT PROPOSAL**

Open 4K RAW Compression

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision # | Author | Revision Date | Comments |
| 1.0 | Gregory Griffith | August 30, 2019 | initiated |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[Project Abstract 4](#_Toc18091814)

[High Level Requirement 4](#_Toc18091815)

[Conceptual Design 4](#_Toc18091816)

[Background 5](#_Toc18091817)

[Required Resources 6](#_Toc18091818)

[References 6](#_Toc18091819)

## Project Abstract

Modern digital video generates a huge data stream that must be stored in some fashion. In most applications this is handled through compression of an RGB image. Many algorithms have been devised over the years, most customized to a specific purpose. Most professional color cameras available at this time utilize a single image sensor. This is done to avoid several artifacts and to simplify optical design. In order to allow a single sensor to record all three colors, the sensor uses a color filter array (CFA) arranged in what is referred to as a "Bayer Pattern". In this configuration, each pixel in the sensor senses only red, green, or blue. The other two values are determined by interpolation within the camera. The parameters used in this process affect all aspects of the image. Over the last decade the industry has acknowledged the creative benefits of preserving the “Bayer Pattern” encoded data as it comes off the image sensor. By manipulating the data in this format, the creator has greater control over the image, as well as allowing for more radical changes to the image before undesired artifacts appear. The recording of a CFA pattern image has become known as recording a "RAW" image. Another benefit of recording this "RAW" image is an immediate reduction in data rate. Reducing the needed bandwidth for a 4K video from GBs/sec, to hundreds of MBs/sec. While this reduction is significant, it still requires a very fast storage medium. In many cases RAW video is needed, but the quality of uncompressed video is not. This allows for manufactures to offer cameras with a good balance between performance and cost.

This project aims to produce an open-source RAW compression module that can be customized and used by other projects producing cameras and video recorders. By targeting an embedded system, we hope to decrease cost over a custom FPGA based solution and improve power efficiency. In addition, the resources of the module provide many opportunities for future features and compression methods.

## High Level Requirement

The product will implement a video capture system that will capture video from an onboard camera, compress it and store it to a storage device. The system will consist of a library that can be used as part of a camera or video recording device. It will provide an API that can be used to configure various parameters of the system. For example, stream source, encoding format, destination. The input will primarily come from the image sensor interface of the processor. Video will be written to either an SSD, PCI-e device, or SD card. Provision for streaming over ethernet may be investigated. While our implementation will use an image sensor, the processor can support other devices such as an HDMI port or an interface to an FPGA to allow for future features and expansion.

## Conceptual Design

The goal of this project will be to produce a library that can compress an HD or 4K RAW video stream. This module will be targeted to perform real-time encoding using the NVIDIA Jetson platform. This is an embedded module providing a high-performance ARM processor and NVIDIA GPU in a mobile platform. The project would read frames from the modules camera and use the OpenJPEG and gpu-jpeg2k libraries to process and compress the images before storing them to disk.

The system would use the modules image processing hardware, along with drivers from the Video for Linux (V4L) library to ingest images into memory. Data would then be distributed to the GPU to be processed and compressed. The library would then receive the compressed video frames and write them to a storage medium. Programming would be done in C++

## Background

At this time there exist very few compression algorithms that can compress RAW video effectively. Most suitable compression methods employ highly tuned predictors to achieve the desired compression ratios. Since the RAW CFA data only records one of the primary colors at each location, the predictors in the algorithms become ineffective.

Several groups have released compressed RAW formats. Cineform RAW, and RED's REDCODE were two of the first. In the last year Apple released PRORES RAW, and Blackmagic Design released their own format. None of these are open formats, in the case of REDCODE the output is in fact encrypted to prevent reverse engineering. Neither Apple nor Blackmagic Design have provided information on decoding their format.

Cineform, now the most open of them all, was eventually acquired by GoPro and used in their cameras, eventually being standardized by the SMPTE as the VC5 codec. In late 2017 GoPro open sourced the SDK and codec, both of which are available in a public GitHub repo. Unfortunately, there is a complete lack of documentation, and the VC5 standard itself is copyrighted and not distributable.

The goal of this product will be to leverage open standards to produce an encoding library that can be used in other open source products. For example, there is a current project to produce an open source cinema camera, the Axiom Beta[1]. This camera currently does not have a recording module.

While there are several open source libraries for full color images that can be leveraged in this product. The product will need to implement several preprocessing steps to prepare the RAW video to be compressed. Most notably the image must be separated into three matrices, one for each color and move to three separate tiles in the image. This results in a green matrix that is twice the width of the red and blue matrices.

In addition, a format for storing the compressed video will need to be designed and implemented.

The simplest implementation for this product would be to use a still image codec such as JPEG2000 or a more modern High Efficiency Image Format (HEIF). The resulting output would be a sequence of compressed still images along with metadata. This has some advantages, such as the ability to pull single frames directly, and simplifies implementation of the product significantly. The trade off is a decreased compression efficiency as compared to a video codec. In a video codec efficiency is improved by using motion estimation to store only the difference between a frame and its predecessors.

One area for experimentation in this project is the choice of codec. In the past JPEG2000 has been used.

There are several other more modern codecs that could be used, for example H.265. A key feature of JPEG2000 is that it is an intra-frame codec. That is each image is encoded in its entirety, the video format is essentially a collection of still images. This is significant for video as it has no motion estimation or compensation components.

Motion estimation becomes difficult in RAW video since the individual color channels are not all square.

If motion compensation and estimation components of the codec are utilized, they can cause undesirable effects in the video. If a video codec is chosen there could be two possible routes. One would be to compress frames individually or find a way to modify the motion compensation to suit RAW video.

Two approaches are described for H.264 video [2], it seems these techniques may be applicable to H.265 as well.

## Required Resources

Initially there will be a significant research effort to determining which methods to use to prepare the image stream for encoding. There will also be quite a bit of research and experimentation with the Jetson platform and CUDA SDKs.

It would be helpful if team members have good C/C++ skills and some experience with CUDA development.

The NVIIDA Jetson is the most significant requirement. The team has one Jetson TX2 kit, it could be beneficial to obtain another that could be setup in a lab on campus and configured for remote access by all team members.

## References

[1] Axiom Beta Project <https://www.apertus.org/axiom>

[2] Doutre, Colin and Panos Nasiopoulos. “Modified H.264 intra prediction for compression of video and images captured with a color filter array.” 2009 16th IEEE International Conference on Image Processing (ICIP) (2009): 3401-3404.