Application Report

Hardware Accelerated Structure From Motion on TDA4VM

Dilshan Godaliyadda, Deepak Poddar, Soyeb Nagori, Do-Kyoung Kwon, and Pramod Swami

ABSTRACT

The ability to accurately map the environment of a vehicle or robot is paramount for even the simplest driver assist or navigation tasks. Structure From Motion, or SFM, is one of the most widely used algorithms that addresses this need, as it allows for accurate mapping with only one camera sensor. This application report describes a hardware accelerated implementation of the SFM algorithm, for Occupancy Grid (OG) Mapping, on the TDA4VM device. The TDA4VM System-on-Chip (SOC) is the first commercially available device in the Jacinto™ 7 family of SoCs, a family of SoCs designed from the ground up specifically for automotive and similar robotics applications. This family of SoCs comprises of two primary variants, the TDA4x family designed for ADAS and similar perception tasks in robotics, and the DRA8x family designed for cloud connected gateway systems. Since the TDA4x family are designed for perception and related analytics, it is equipped with hardware accelerators and general-purpose processing cores to which algorithms such as SFM can be seamlessly mapped. This mapping as it pertains to SFM will be explored in this application report.

Table of Contents

Trademarks

1 Introduction

Whether it is a simple task such as lane assist or blind spot detection, or a more complex task such as autonomous navigation, understanding the surroundings of a vehicle or robot is vital for success, and thereby safety. Vehicles and robots perceive their environment by converting data captured by sensors such as, RADARs, LiDARs and Cameras in to a format that can be consumed by the vehicle's decision-making engine. Light Detection and Ranging (LiDAR)-based maps tend to be the most accurate, however, they are typically cost prohibitive for most vehicles or robots. Therefore, RADAR and camera-based solutions tend to be more widely used.

The SFM algorithm is one of the more widely used algorithms for camera-based mapping. The SFM algorithm by itself outputs a point-cloud (a set of points extracted from surrounding objects), which can then be consumed by some type of mapping algorithm. The application described in this article feeds the point-cloud to an Occupancy Grid mapping algorithm to generate a map of the surroundings.

In automotive and robotics applications, the steps of receiving sensor data, converting the data to a usable format, and prescribing actions based on the perceived environment, are typically performed on an embedded platform. The Jacinto 7 TDA4x family of high-performance SoCs by Texas Instruments are designed from the ground up particularly to address the varied algorithmic needs of the automotive, industrial and robotics markets. The Structure From Motion, or SFM algorithm is one such algorithm around which the device was designed. As a result the key computational blocks of the algorithm map seamlessly to either hardware accelerators or general purpose processing cores on the TDA4VM device. This article describes the SFM based OG mapping algorithm, the TDA4VM device, and how the algorithm maps to the device to enable a high-fidelity real-time map of the environment, before showing some example implementations on the device with corresponding outputs.

2 What is Structure From Motion?

In Computer Vision, the position of an object with respect to a vehicle is ascertained using images from two cameras, mounted on known disparate locations, looking at the object in question. In particular, key points from the object in both images are extracted and matched, and then using a process known as Triangulation the locations of the points that make up the object are deciphered. The process of distinguishing the position of a point in space using two cameras is known in the Computer Vision community as Stereo Vision, or Stereo Depth Estimation, and the set of points generated from all the correspondences in the two images is referred to as a point-cloud. Even as Stereo Vision is widely used by the automotive and robotics communities, it comes at a high system cost in terms of both dollars and image processing requirements, because it requires two high-precision cameras, capturing images at a relatively high frequency.

In contrast, Structure From Motion, or SFM, is an algorithm that can generate a point-cloud from a single camera in motion. As the name implies, in SFM, we have one camera which due to motion is in two distinct locations at two consecutive time instances, which effectively is the same as placing two cameras in distinct locations, given the objects in the frame have not moved between the two time instances, and given we know the relative motion of the camera. Thus, one can effectively use the same theory as in Stereo Vision to generate a point-cloud, from just one camera.

SFM algorithms come in two primary flavors, traditional Computer Vision based and Deep Learning based. Even though both flavors can be executed on TDA4VM, in this document, the focus in on the former, an algorithm based on traditional Computer Vision techniques. The point cloud generated by the SFM algorithm needs to then be utilized to generate a map of the surroundings, and in the application described here a 2D OG mapping approach is utilized for the mapping task.

3 Introduction to Occupancy Grid Mapping

Occupancy Grid Maps are a widely used method to represent the environment surrounding a vehicle or robot, because they can be consumed by a variety of ADAS applications ranging from parking, to obstacle identification, to curb detection, and be stored efficiently. An OG map is a representation of a vehicle's surroundings in the form of a 2-Dimensional (2D) or 3-Dimensional (3D) grid. Each grid cell in the map has a corresponding state, for example *occupied, free or unknown*, computed from information received from sensors such as RADARs, Cameras, or LiDARs. A simple illustrative example of an OG map is shown in Figure 3-1.

OG maps can either be centered on the vehicle or robot, or centered on an arbitrary frame of reference. The former is known as an *ego-centric* OG map and the latter as a *world-centric* OG map. OG maps are further categorized as accumulated or instantaneous, based on whether or not information from previous frames is used.

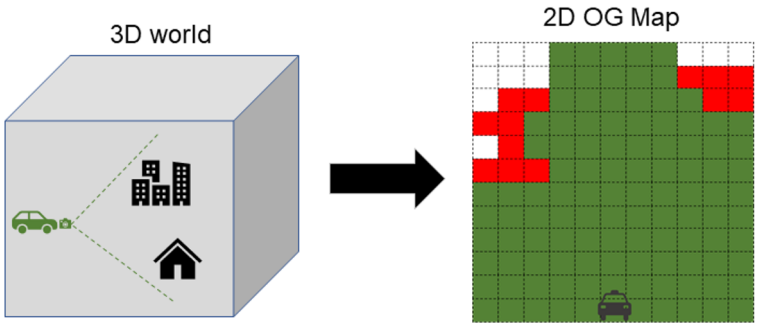

Figure 3-1. Left: Illustration of World; Right: Illustration of an OG Map (Green cells empty, Red cells occupied, and white unknown)

This application report describes a *world-centric* 2D OG map constructed using a point-cloud generated from measurements acquired by a monocular camera. This OG map has three states – *occupied*, *free* or *unknown*. The primary focus of this application report will be on an instantaneous OG mapping algorithm. Then, towards the latter part of the article, an accumulated OG mapping application that is also included with the Software Development Kit (SDK) that accompanies the TDA4VM device is briefly described. The next section describes how to take a point-cloud as input and generate an OG map based on the point-cloud and the location of the vehicle/robot.

4 From Point-Cloud to OG Map

To create a world-centric OG Map, first each point in the point-cloud generated by the SFM algorithm, is placed in a world-centric coordinate system using vehicle location data from the Inertial Navigation System, or INS, and sensor calibration data. Sensor calibration data includes the relative positions of the different sensors with respect to each other. Then, each point is classified as either originating from the ground (*ground point*) or from an object (*object point*). Next, the state of each cell in the OG map is determined using the world-centric point-cloud as follows -- if a predetermined number of *object points* are found in a cell, it is deemed *occupied*; if a predetermined number of *ground points* are found in a cell, it is deemed *free*; if neither criterion is met it is deemed *unknown*.

5 Algorithm Flow: SFM-Based OG Mapping

The algorithm flow of the SFM based OG Mapping algorithm described in this application report is shown in Figure 5-1.

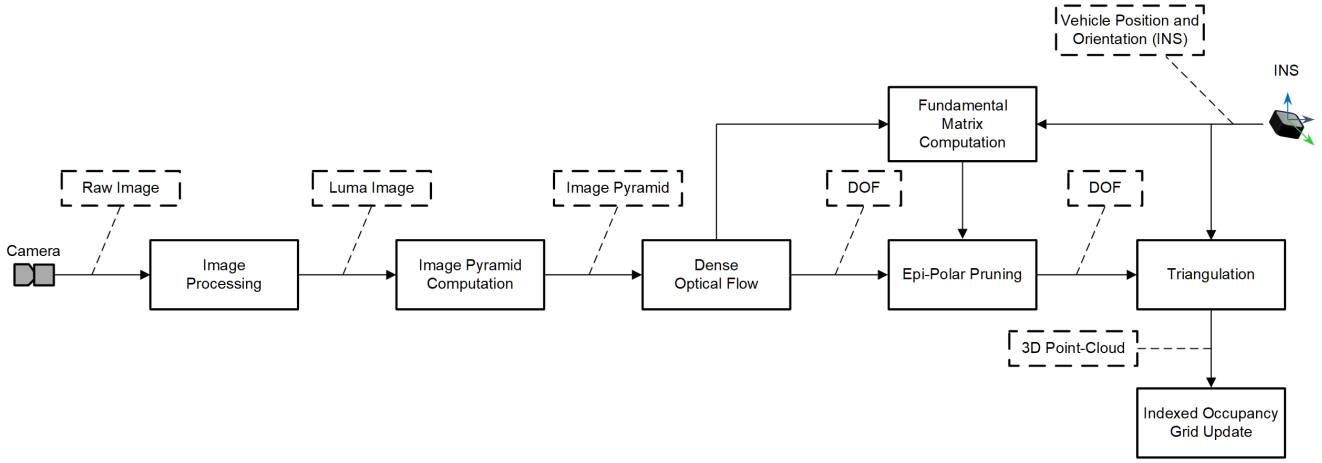

Figure 5-1. SFM-Based Occupancy Grid Mapping Algorithm Flow

It is important to note that some of the subtasks such as the Dense Optical Flow (DOF) block, Image Pyramid Formation and Image Processing need to be performed on hardware accelerators (HWAs) because without hardware acceleration, these processing intensive tasks cannot be performed in real-time.

6 Algorithm Flow: SFM-Based OG Mapping on TDA4VM

The TDA4VM SoC, which was designed with SFM and other popular Computer Vision tasks necessary for ADAS and robotics in mind, is unsurprisingly uniquely suited for the SFM algorithm. Figure 6-1 shows the TDA4VM device, and Figure 6-2 shows the algorithm flow in Figure 5-1 mapped on to TDA4VM. The Vision Pre-processing Accelerator, or VPAC, mentioned here contains modules to accelerate different image pre-processing sub tasks such as tone mapping, noise filtering, lens distortion correction, and so forth. The Depth and Motion Processing Accelerator, or DMPAC, contains two modules, a Stereo Disparity Engine, or SDE, to accelerate stereo depth estimation, and a Dense Optical Flow Engine, or DOF engine, to accelerate DOF. For more information on the TDA4VM device, see https://www.ti.com/product/TDA4VM.

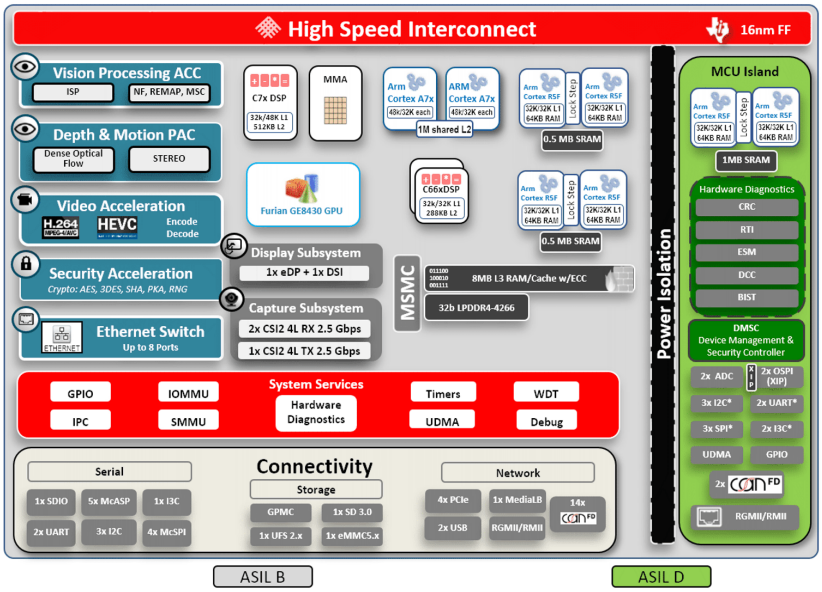

Figure 6-1. TDA4VM Diagram

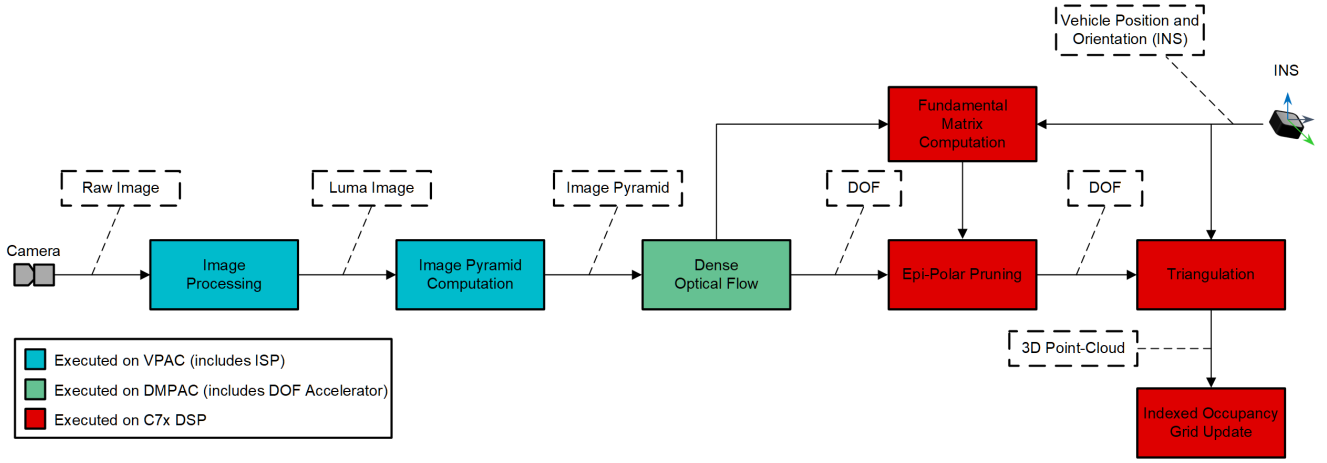

Figure 6-2. SFM-Based Occupancy Grid Mapping Algorithm Flow on TDA4VM

Table 6-1 shows how the sub-tasks of the SFM OG mapping algorithm map are tabulated to the different processors in the TDA4VM SoC.

Table 6-1. SFM OG Mapping Algorithm, Sub-Task Mapping to TDA4VM

Step	TI Processor Silicon IP
Raw Image Capture	Imaging Sensor
Transmit image to Processor	CSI-2 port
Image Processing to form Luma image	Vision Pre-processing Accelerator (VPAC)
Image Pyramid Formation	Vision Pre-processing Accelerator (VPAC)
Dense Optical Flow Computation	Depth and Motion Processing Accelerator (DMPAC)
Epi-Polar Pruning	C7x DSP
Fundamental Matrix Computation	C7x DSP
Triangulation	C7x DSP
Indexed OG map Update	C7x DSP

Without acceleration through dedicated HWAs, some of these steps are computationally prohibitive. For example, take the DOF portion of the algorithm. For a 2MP camera input, the C7x DSP consumes 2000 Mega Cycles Per Second (MCPS) to generate 2000 key points. Whereas, the DOF HWA only consumes 394 MCPS. Table 6-2 summarizes the results.

Table 6-2. DOF Computation Comparison on C7x DSP Versus DOF HWA for a 2MP Camera Input

TI Processor Silicon IP	Mega Cycles Per Second (MCPS)
DOF HWA on DMPAC	394
C7x DSP	2000

Furthermore, since all sub-tasks in this algorithm can be performed using the DSPs and HWAs, the general-purpose ARM cores and Deep Learning Accelerators in the TDA4VM device can be reserved for other applications.

Next, two example implementations of SFM OG mapping are described. Both of these applications are included in the SDK that accompanies TDA4x devices. The first, is intended for a vehicle operating on the road, and therefore is designed for input data from a camera mounted in the front of the vehicle. The second, is for automatic parking, and therefore uses input from a camera mounted on the side of the vehicle.

7 First Example Implementation on TDA4VM

This application runs on saved images and INS data, and therefore skips the camera sensor to VPAC leg of the diagram above. Documentation on how to run the application as well as all the necessary libraries can be found in the SDK, and are listed in Section 9. Application specific documentation can be found here .

A sample output of this application is shown in Figure 7-1. In the top of diagram, we show the output of Dense Optical Flow and in the bottom the output of Sparse Optical Flow (SOF). From this diagram, it is clear why DOF is preferred over SOF for applications such as OG Mapping, as oncoming cars cannot be detected with a sparse point cloud. The experiment runs on simulated data generated using the Carla Simulator 0.9.9.4 [1], with a virtual camera mounted on the front of the vehicle. Here, free cells are marked green, occupied red and unknown yellow. This example runs at 30 fps on 0.5 MP camera data with less than 60% C7x DSP utilization to generate 20,000 3D key points. Although the ISP portion is skipped in this application, the ISP has the ability to process data at 600 MP/s and, therefore, can accommodate a 0.5 MP monocular camera at 30 fps. The user can request a Data Sheet with precise information about different core utilization here.

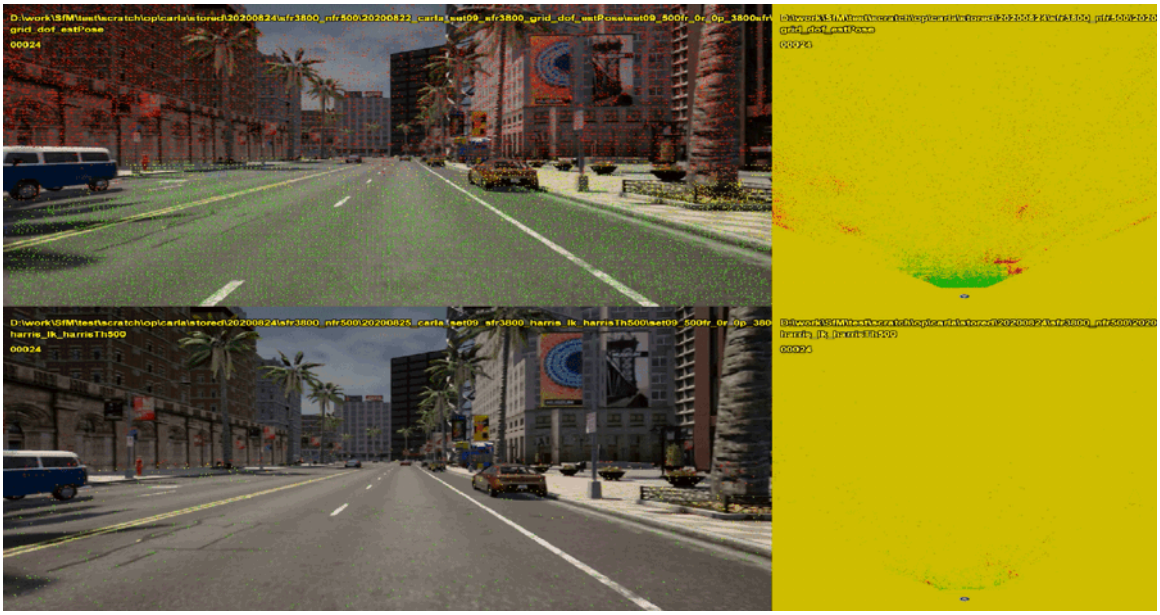

Figure 7-1. Example Output From SFM OG Mapping Application With DOF [top] and SOF [bottom]

Examples in the SDK, including the SFM OG mapping example, are made available to customers to shorten their development time. If customers prefer their own pipeline or a modified pipeline, these examples can be used as reference or as a starting point.

8 Second Example implementation on TDA4VM

This SFM based OG mapping application is available in the SDK under the PTK demos section. The documentation for this algorithm can be found here. This particular application constructs a world-centric accumulated 2D OG map, as opposed to the instantaneous map created by the above application. This algorithm uses almost an identical chain to the one described above, with minor differences. A sample output of this application is shown in Figure 8-1. Here, the camera is mounted on the side of the vehicle and also runs on saved data captured in a parking lot inside TI. Here, free cells are marked green, occupied red and unknown black.

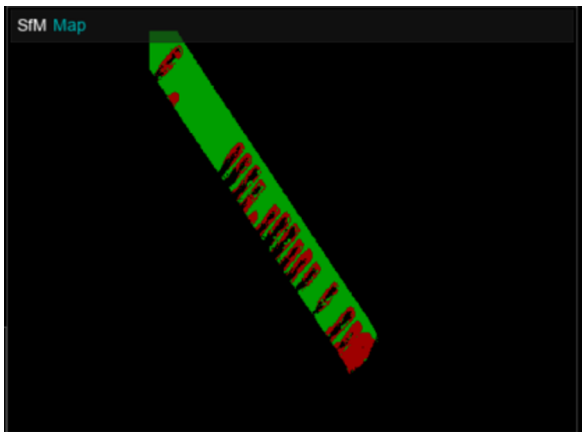

Figure 8-1. Example Output From SFM OG Mapping Application

9 References

1. Dosovitskiy, Alexey, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. "CARLA: An open urban driving simulator." In Conference on robot learning, pp. 1-16. PMLR, 2017.

Table 9-1. Important Links

Reference Name	Link
TDA4VM product information:	https://www.ti.com/product/TDA4VM
Evaluation Board Purchase/Request	Common Board: https://www.ti.com/tool/J721EXCPXEVM TDA4VM Processor SOM: https://www.ti.com/tool/J721EXSOMXEVM
Download SDK	https://www.ti.com/tool/PROCESSOR-SDK-DRA8X-TDA4X
SDK Documentation	https://software-dl.ti.com/jacinto7/esd/processor-sdk-rtos-jacinto7/latest/exports/docs/psdk_rtos/docs/user_guide/index.html
Jacinto 7 Video Training series	https://training.ti.com/jacinto7-platform
Technical Support from E2E Processor Forum:	https://e2e.ti.com/support/processors/f/791
PTK API Guide (library used in second application)	https://software-dl.ti.com/jacinto7/esd/processor-sdk-rtos-jacinto7/latest/exports/docs/perception/docs/ptk_api_guide/index.html

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265