

## COMPASS Sensor Modelling Application

R H Sutherland and M I Cowan  
INSYS Ltd.  
Reddings Wood, Ampthill, Bedfordshire, MK45 2HD

### Abstract

*This paper discusses the requirement and design objectives for a proposed software application (Compass) to model electro-optical (EO) sensors and to simulate their image degradation effects. The application is aimed at two types of user: the analyst, who simply wants to create degraded imagery for analysis, and the sensor designer or developer, who wants a tool to develop new sensor models without needing specialist programming skills. The solution is aimed at providing a high degree of flexibility for the sensor developer while retaining computational speed and ease of use.*

*Keywords: Compass, CameoSim, EO-Sensors, Hyperspectral, Polarimetry*

### Introduction

Electro-optical sensors play an increasingly important role in defence and there is a need for more capable sensors to satisfy the demand for better intelligence. New sensor concepts such as multispectral, hyperspectral and polarimetric types have appeared in recent years. Sensors must be designed, developed and tested and then be submitted for trials before obtaining approval for use. During the initial stages of development sufficient confidence in the design needs to be obtained before committing to the later and more expensive phases, particularly trials. This is achieved by developing computer models of the sensors and simulating and assessing the effect they have on imagery.

Imagery may be obtained from trials observations or it may be generated by computer simulation.

Trials imagery is clearly realistic, but trials are costly, are not repeatable and it is difficult to achieve the desired weather, seasonal and lighting conditions. Trials imagery also contains sensor artefacts, which may not be representative of the sensor under consideration; this makes it

difficult to manipulate the imagery, for instance, by “patching” objects into the scene.

Computer-generated synthetic imagery has become increasingly popular in recent years. Many people in the EO field use *CameoSim*<sup>1</sup> to produce computer-rendered synthetic images. *CameoSim* is based on physics and the imagery it produces is radiometrically accurate. Synthetic imagery can be created for a range of scenes, weather conditions, time of day, time of year and viewpoint. It is fully controllable by the user and has the advantage of repeatability. *CameoSim* imagery includes atmospheric effects and accurately predicts the image of the scene as presented to the optics of a sensor.

*CameoSim* allows images to be generated for a user-defined set of sub-bands within any given waveband over the range 0.4 - 20 $\mu$ m. This allows the user to generate imagery of the type that would be seen, for instance, in a two-colour infrared sensor or

---

<sup>1</sup> *CameoSim* ([www.cameo-sim.com](http://www.cameo-sim.com)) is a physics-based application developed by INSYS Ltd. ([www.insys-ltd.co.uk](http://www.insys-ltd.co.uk)) for the generation of synthetic imagery. It is used by the EO community both in the UK and overseas.

a dispersive hyperspectral sensor. Work is currently in progress to develop *CameoSim* for polarimetric imagery, which it is hoped will allow for the assessment of future battlefield and research EO sensors.

A user working with imagery needs to be able to apply sensor effects and to generate realistic “degraded” imagery representing either the final sensor output, or imagery at any intermediate point in the degradation process. *Compass*<sup>2</sup> has been designed to fulfil this need by providing a tool which is versatile, flexible and easy to use by a sensor specialist who does not wish to or cannot afford to devote a great deal of time to computer programming.



Figure 1 *CameoSim* image in the visible band

## Compass

*Compass* is planned as a stand-alone application, but it will be fully capable of operating with *CameoSim* imagery. *Compass* is aimed at two different types of user, termed the analyst (User\_1) and the sensor model developer (User\_2). User\_1 would perceive *Compass* as an application for generating degraded sensor imagery using approved and complete sensor modules. User\_2 would view *Compass* as a

<sup>2</sup> *Compass* is an acronym for Common Optical Modelling Platform for Analysis of Sensor Systems

tool for building and testing new sensor models or for modifying existing ones. User\_2 will want the ability to conduct parametric studies, sensitivity studies and trade-off studies on sensor parameters at system, sub-system and component level and to examine the impact of physical effects within a sensor on its performance.

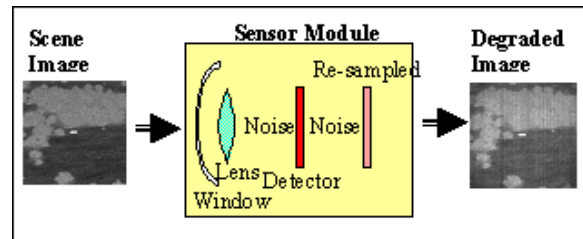


Figure 2: Example Components of the Sensor Module

User\_1 would view the validated sensor module(s) as an integral part of the application and would require and expect reasonable computational speed. User\_2 requires the flexibility to modify or build sensor modules. The design of *Compass* meets the requirements for flexibility and computational speed, which often impose conflicting constraints on the design, through its use of a compiled library combined with a simple scripting language.

## Functional Requirements

A fundamental requirement for *Compass* is that it should have a compiled library of functions that will provide the building blocks from which models of a wide variety of sensor types may be built. In the first instance, five examples have been chosen as a basis for developing the software. These (listed below) vary in complexity.

### 1. Simple Case

This example represents the simplest possible case, a sensor defined by nominal parameters such as the field of view, the focal length, aperture diameter, and transmission of the optics; the number, size and pitch of the detector pixels; the noise level and the A/D (grey scale) conversion.

## 2. Noise and Non-Uniformity Case

This example is similar to Example 1 but allows a more detailed level of modelling. Photon noise, temporal and spatial (fixed-pattern) noise are included as well as filters and a non-linear detector response, saturation, gain/offset control and digitization. Convolution is carried out twice to distinguish between pre- and post-detector MTF. The first stage represents the application of the optics MTF. The second stage represents the processes behind the detector and is applied to down-sampled images.

## 3. Wavelength-Dependence Case

This is a development of Example 2 to include wavelength-dependence. A hyperspectral stack of images will provide the wavelength-dependent input imagery. All of the functionality of Example 2 is included but the spectral characteristics of the filters, the detector response, the MTF (dispersion) are added. This allows modelling at a high level of fidelity.

## 4. Detailed Sensor Case

This is a development of Example 3 and is intended to show how a model can be created to examine physical effects in close detail. This example includes optical effects such as variation of the MTF with field angle (also tangential and sagittal MTF), vignetting and shading, non-linear and non-isoplanatic detector responsivity, non-uniformity, non-uniformity correction (NUC) and persistent effects in the detector arising from NUC, as well as dead pixel replacement (also “blinkers” and “flashers”).

## 5. Hyperspectral Sensor Case

This example is different from Examples 1 to 4 in that it represents the processes in a dispersive hyperspectral imager and the generation of “hypercubes”.

The processes involved are as follows: (1) Parsing of hyperspectral image data and gathering of calibration data, (2) Image formation (including blurring and sampling), (3) A/D conversion (4) Calibration and (5) Accumulation.

## Proposed Solution

The *Compass* architecture, shown in Figure 3, includes a powerful mathematical computational engine particularly tuned to matrix operations.

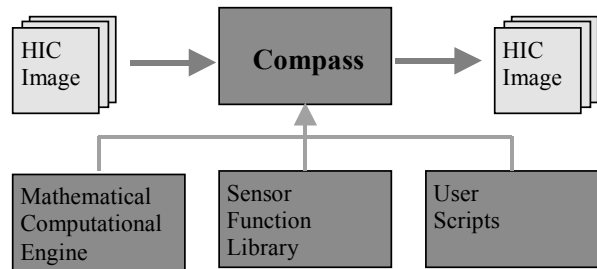


Figure 3: *Compass* Architecture

*Compass* must accept a temporal stack of hyperspectral image cubes (HIC) and output the degraded images in the same format [it is possible that the image cube may be smaller on output as a consequence of some intermediate integration of the image sub-bands]. Many of the computations used are the same regardless of the sensor being modelled. Those that are common will be developed as a set of functions and made as general purpose as possible to widen their use. Three sets of such functions have been identified and classified as:

- Set 1 General functions
- Set 2 Specific functions
- Set 3 Detailed functions

These functions will be supplied in a compiled library to facilitate computational speed.

A high-level scripting facility is provided to allow the user to develop new functionality. He or she will be able to call up any of the functions from the sensor library in addition

to having access to functions provided by the mathematical engine. The scripted language is simple and high-level and no specialist coding skills are required to use it. This provides User\_2 with a very flexible productive system that will allow specific sensor models to be developed, as well as providing a powerful tool for parametric analysis.

### Scripting

An example of creating scripts is illustrated in Figure 4 for a sensor model that satisfies the requirements of Example 1. The symbols represent functions from the sensor model library. The sensor script 'SensorA1' is called from a master script that deals with loading and saving the image data.

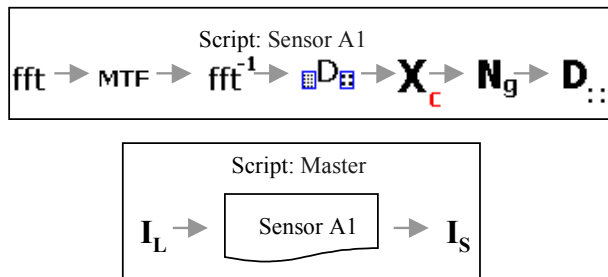


Figure 4: Sensor Scripts to Satisfy Example 1

Scripts can be nested as indicated above; this allows the user to model the sensor in a modular fashion. An alternative to the script is shown in Figure 5, where the sensor is divided into two components: the optics and the detector. A separate script is written for each component and called from a higher-level script.

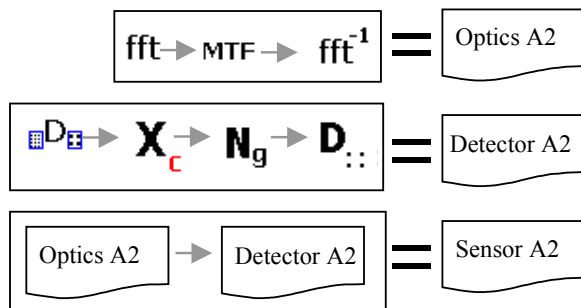


Figure 5: Alternative Sensor Scripts to Satisfy Case A

The combination of nested scripting with the mathematical and sensor function

libraries provides a rich, powerful and flexible system for building, modifying and testing sensor models. Furthermore the approach lends itself to prototyping and incremental development. Once a model is developed and tested in a script(s) it can be converted into compiled code to increase speed and added to a sensor module library (for simplicity this was not shown in the architecture of Figure 2).

The balance between computational speed and flexibility is achieved through compiled libraries for speed, and scripts for flexibility.

### Requirements Capture

A draft *Compass* requirement specification was presented at a number of meetings with Dstl and several UK Defence companies. Feedback was obtained via comments on the requirement specification and by a questionnaire. The questionnaire responses showed a marked preference for Windows PC as the hardware platform and for Matlab as the mathematical engine. INSYS Ltd are indebted to the following for their participation in the requirements capture programme.

- BAE Systems ATC (Filton)
- Dstl (Farnborough, Malvern)
- Thales Optronics
- QinetiQ (Farnborough, Malvern)

### Current Position and Future Development

*Compass*, at the time of writing, is at the top-level design stage. A phased development approach has been taken with Phase 1 shown in Figure 6. The output from this phase will be *Compass* with a limited Graphical User Interface (GUI), but with a full implementation of the General and Specific function libraries as well as a full Generic and Specific sensor module. The labelled markers in Figure 6 identify outputs in Phase 1 in which 1 and 3 are appropriate to User\_2 and 2 and 4 are

appropriate to User\_1. Phase 2 will cater for the detailed sensor functions, the GUI and any requirements for interfaces into other relevant applications.

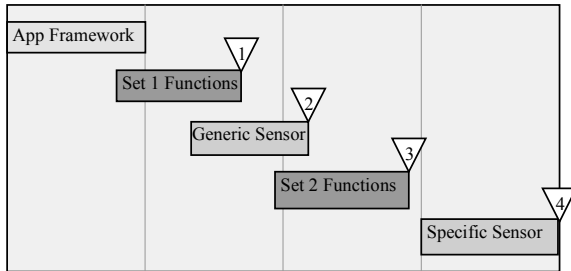


Figure 6: Development Programme for Phase 1

It is emphasized that *Compass* is a stand-alone application. It may be used as a means of processing imagery generated by *CameoSim* but it is not limited to that use. It will be capable of handling imagery from other sources (trials data as well as other synthetic imagery) and may be used to perform other sensor calculations such as the familiar calculations of detection, recognition and identification ranges using MRTD. It may also be used to aid in validation and as a medium for interfacing with other commercial codes, such as optical design software, CFD and target signature codes.

It has been mentioned that *CameoSim* is being developed for polarimetric imagery. *Compass* will be developed to allow modelling of polarimetric sensors.

As *Compass* develops, the user-friendliness will be enhanced by the addition of GUIs tailored to a variety of tasks.

In the long run, *Compass* will become an indispensable aid to anyone involved in the assessment, design and development of EO sensors.