

GSFC JPSS CMO
April 24, 2017
Released

Joint Polar Satellite System (JPSS) Ground Project
Code 474
474-00085

Joint Polar Satellite System (JPSS) Operational Algorithm Description (OAD)

Document for VIIRS Cloud Cover/Layers (CCL) and Generate Cloud EDR (GCE) Software

For Public Release

The information provided herein does not contain technical data as defined in the International Traffic in Arms Regulations (ITAR) 22 CFC 120.10. This document has been approved For Public Release to the NOAA Comprehensive Large Array-data Stewardship System (CLASS).



Goddard Space Flight Center
Greenbelt, Maryland

National Aeronautics and
Space Administration

**Joint Polar Satellite System (JPSS)
Operational Algorithm Description (OAD) Document for
VIIRS Cloud Cover/Layers (CCL) and Generate Cloud
EDR (GCE) Software
JPSS Electronic Signature Page**

Prepared By:

Ray Godin
JPSS Data Products and Algorithms EDR Lead
(Electronic Approvals available online at (https://jpssmis.gsfc.nasa.gov/mainmenu_dsp.cfm))

Approved By:

Gilberto Vicente
JPSS Ground Project Algorithm Integration Team (AIT) Manager
(Electronic Approvals available online at (https://jpssmis.gsfc.nasa.gov/mainmenu_dsp.cfm))

**Goddard Space Flight Center
Greenbelt, Maryland**

Preface

This document is under JPSS Ground Algorithm ERB configuration control. Once this document is approved, JPSS approved changes are handled in accordance with Class I and Class II change control requirements as described in the JPSS Configuration Management Procedures, and changes to this document shall be made by complete revision.

Any questions should be addressed to:

JPSS Configuration Management Office
NASA/GSFC
Code 474
Greenbelt, MD 20771

Change History Log

Revision	Effective Date	Description of Changes (Reference the CCR & CCB/ERB Approve Date)
Original	06/03/2011	474-CCR-11-0100: This version baselines D39590, Operational Algorithm Description Document for VIIRS Cloud Cover/Layers (CCL) and Generate Cloud EDR (GCE) Software, Rev B dated 03/17/2010 as a JPSS document, version Rev -. This is the version that was approved for NPP launch. Per NPOESS CDFCB - External, Volume V – Metadata, doc number D34862-05, this has been approved for Public Release into CLASS. This CCR was approved by the JPSS Algorithm ERB on June 3, 2011.
Revision A	01/18/2012	474-CCR-11-0266: This version baselines 474-00085, Joint Polar Satellite System (JPSS) Operational Algorithm Description (OAD) Document for VIIRS Cloud Cover/Layers (CCL) and Generate Cloud EDR (GCE) Software, for the Mx6 IDPS release. This CCR was approved by the JPSS Algorithm ERB on January 18, 2012.
Revision B	10/09/2012	474-CCR-11-0627: This version authorizes 474-00085, Joint Polar Satellite System (JPSS) Operational Algorithm Description (OAD) Document for VIIRS Cloud Cover/Layers (CCL) and Generate Cloud EDR (GCE) Software, for the Mx6.1 – 6.3 IDPS releases. Includes ECR-ALG-0035 which contains Raytheon PCR030752, OAD: Implement 474-CCR-12-0402 (Add Mirror Side Information to VIIRS CCL/GCE OAD) (ADR 4703), updated Table 14 and various references.
Revision C	05/14/2013	474-CCR-13-0948: This version authorizes 474-00085, JPSS OAD Document for VIIRS CCL & GCE EDR Software, for the Mx 7.0 IDPS release. Includes Raytheon PCR032720; 474-CCR-13-0916/ECR-ALG-0037: Update applicable OAD filenames/template/Rev/etc. for Mx7 Release.
Revision D	11/06/2013	474-CCR-13-1288: This version authorizes 474-00085, JPSS OAD Document for VIIRS CCL & GCE EDR Software, for the Mx 8.0 IDPS release. Includes administrative changes authorized by interoffice memo and Raytheon PCR034611; OAD: PRO: 474-CCR-13-1032: VIIRS Generate Cloud EDR (GCE) ADR7037 - Remove Sunlint Over Land from Cell-level Overall Quality Calculation, in Table 18.
Revision E	03/13/2017	474-CCR-17-3243 (ECR-CGS-0734): This version authorizes 474-00085, JPSS OAD Document for VIIRS CCL & GCE EDR Software, for the Block 2.0 IDPS release. Includes Raytheon PCR045678: Block 2.0: PRO: OAD: CCR: 474-CCR-15-2444: General OAD Clean-up CCR/PCR, affects all 35/37 OADs. All sections and tables may be affected.



NATIONAL POLAR-ORBITING OPERATIONAL ENVIRONMENTAL SATELLITE SYSTEM (NPOESS)

OPERATIONAL ALGORITHM DESCRIPTION DOCUMENT FOR VIIRS CLOUD COVER/LAYERS (CCL) AND GENERATE CLOUD EDR (GCE)

**SDRL No. S141
SYSTEM SPECIFICATION SS22-0096**

**RAYTHEON COMPANY
INTELLIGENCE AND INFORMATION SYSTEMS (IIS)
NPOESS PROGRAM
OMAHA, NEBRASKA**

**Copyright © 2005-2011
Raytheon Company
Unpublished Work
ALL RIGHTS RESERVED**

Portions of this work are the copyrighted work of Raytheon. However, other entities may own copyrights in this work. Therefore, the recipient should not imply that Raytheon is the only copyright owner in this work.

This data was developed pursuant to Contract Number F04701-02-C-0502 with the US Government under subcontract number 7600002744. The US Government's rights in and to this copyrighted data are as specified in DFAR 252.227-7013, which was made part of the above contract.

IAW DFAR 252.227-7036, Raytheon hereby declares that, to the best of its knowledge and belief, the technical data delivered under Subcontract No. 7600002744 is complete, accurate, and complies with all requirements of the Subcontract.

TITLE: NATIONAL POLAR-ORBITING OPERATIONAL ENVIRONMENTAL SATELLITE SYSTEM (NPOESS) OPERATIONAL ALGORITHM DESCRIPTION DOCUMENT FOR VIIRS CLOUD COVER/LAYERS (CCL) AND GENERATE CLOUD EDR (GCE)

APPROVAL SIGNATURES:



17 Mar 2010

Stephen E. Ellefson
ING/PRO Lead

Date



17 Mar 2010

Gabriela Ostler
Mission Assurance and Enterprise Effectiveness (MAEE)

Date

Northrop Grumman Space & Mission Systems Corp.
 Space Technology
 One Space Park
 Redondo Beach, CA 90278



**Engineering & Manufacturing Development (EMD) Phase
 Acquisitions & Operations Contract**

CAGE NO. 11982

**Operational Algorithm Description Document for the
 VIIRS Cloud Cover/Layers (CCL) and Generate
 Cloud EDR (GCE) Software**

**Document Number: D39590
 Revision: C5**

Document Date: Nov 14, 2011

PREPARED BY:

Eric Wong <i>Date</i> <i>AM & S Algorithm Lead</i>	 Paul D. Siebels <i>Date</i> <i>IDPS PRO SW Manager</i>
	17 Mar 2010

ELECTRONIC APPROVAL SIGNATURES:

Roy Tsugawa <i>Date</i> <i>A&DP Lead & ACCB Chair</i>	 Stephen E. Ellefson <i>Date</i> <i>IDPS Processing SI Lead</i>
	17 Mar 2010
Bob Hughes <i>Date</i> <i>A&DP Deputy & ARB Chair</i>	

Prepared by
 Northrop Grumman Space & Mission Systems Corp.
 Space Technology
 One Space Park
 Redondo Beach, CA 90278

Prepared for
Department of the Air Force
 NPOESS Integrated Program Office
 C/O SMC/CIK
 2420 Vela Way, Suite 1467-A8
 Los Angeles AFB, CA 90245-4659

Under
Contract No. F04701-02-C-0502

This document has been identified per the NPOESS Common Data Format Control Book – External Volume 5 Metadata, D34862-05, Appendix B as a document to be provided to the NOAA Comprehensive Large Array-data Stewardship System (CLASS) via the delivery of NPOESS Document Release Packages to CLASS.

Northrop Grumman Space & Mission Systems Corp. Space Technology One Space Park Redondo Beach, CA 90278		 	
Revision/Change Record		Document Number	D39590
Revision	Document Date	Revision/Change Description	Pages Affected
--	12-23-04	Initial Release to NGST for P1187-SW-I-017; TM # NP-EMD.2005.510.0077 already implemented.	All
A1	1-21-05	<u>Initial PCIM Release. ECR A046</u> Release as Document D39590, SPCR 736 Correct various typos and formatting nits Update to referenced documents Update to CCL Processing chain diagram to use Corrected IPs Update CCL/GCE Input table to use Corrected Ips State in CCL Cloud Cover and CCL EDR output tables that Total Cloud Cover output is cloud fraction <i>summed</i> over all layers instead of <i>averaged</i> Add comment regarding future implementation of GCE quality flags, add TBD addressing this issue to TBD table Add content to Test Procedure and Results sections for CCL and GCE.	All various 1 3 5 9, 12 1, 19 24, 27
A2	6-12-05	Implementation of quality flags and function arguments per SPCR ALG 629.	All
A3	6-12-05	Implementation of CCL logic based on AER doc P1187-SW-I-017_CCL_and_GCE-OAD.doc SPCR ALG 843.	21
A4	3-22-06	Reflects Raytheon-Omaha's initial edits for Science To Operational (Sci2Ops) Code Conversion Process, adding Raytheon coversheet, title/signature page, etc. Updated coversheet copyright; replaced old Unit Test with the latest 16Feb06 version; did other edits per Omaha QA comments posted for I-P-O CUTPR and TM # NP-EMD.2006.510.0007. Removed Sections 2.3.6.15 through 2.3.6.20 per the PRO SW engineer working the GCE algorithm. Renamed Figures 1 and 2 to differentiate CCL versus GCE processing chain then updated List of Figures.	All
A5	12-6-06	Updated for error handling and data quality, combining 2 ccl outputs into 1, Figure 1, Tables 1 and 10, Sections 2.2, 2.3.4, and 4.4.	3,5,8,26,36
A6	1-6-07	Updated per EH DQ DDPDR reviewer comments.	All
A7	1-19-07	Updated per Optimization DDPDR review comments.	All
A8	2-15-07	Final update of CCL portion per new OAD template format.	All
A9	2-16-07	Delivered to NGST.	All

Northrop Grumman Space & Mission Systems Corp. Space Technology One Space Park Redondo Beach, CA 90278		 	
Revision/Change Record		Document Number	D39590
Revision	Document Date	Revision/Change Description	Pages Affected
A10	4-26-07	Updated for TM # NP-EMD.2005.510.0038. Updated Section 2.1.2 to include information regarding reading the last scan from the previous granule and the first scan of the next granule and supply this information to the algorithm for processing. Removed EdgeHandlingMode row from Table 11 since field was not used. Updated Table 12, added missing value.	All
A11	5-30-07	Updated per GCE sci2ops error handling and data quality. Removed granule flags from EDRs (they are now posted to metadata). Added summaries for GCE functions. Updated per code completion review comments.	All
A12	6-14-07	Updated GCE portion to new format: updated tables including listing quality flags at the bit-level; modified figures; updated function descriptions and removed input/output tables for functions; added tech memos to reference doc table. Updated CCL portion for output product names.	All
A13	6-15-07	Delivered to NGST.	All
A14	6-26-07	Update scale and offset to Float32.	27-32
A15	7-12-07	Added cell_copy_out() per fix in PCR013942.	18,20
A16	8-9-07	TM NP-EMD.2007.510.0034 is implemented in CCL in B1.5.	All
A17	8-20-07	Removed scaling coefficients from GCE Coefficients table (Table 17), scaling of products is now performed in common code (i.e. ProCmnProductDictionary) instead.	25
A18	10-24-07	Made corrections for flipped layer assignments. Delivered OAD to NGST.	All
A19	11-9-07	Added spacecraft position, velocity, and attitude to geolocation output. Responded to ECR A-122 PR comments.	14-15
A20	12-17-07	ECR A-103 Updates: added many new fields to the output geolocation structure and added a new method assignScanLevelGeolocationData().	All
A21	7-10-08	Accepted track changes. Rewrote DQN text.	All
A22	7-23-08	Prepared for delivery to ACCB. Updated Acronym list and references. Implemented new cover sheet from NGST.	All
A	9-2-08	ECR A-167. Incorporated interim changes and addressed TIM comments.	All
B1	10-20-08	Modify Cloud Cover Layers Algorithm (CCL) to read the new cloud mask bits and fix software bugs identified by IDPS(tech memo -NP-EMD-2008.510.0017). Also removed the flipped layer code from CCL. Test results were delivered to NGST (GR_SE_EMD_16294)	3 and 19
B2	11-09-08	Modified GCE algorithm to re-implement TM NP-EMD.2007.510.0034.	37

Northrop Grumman Space & Mission Systems Corp. Space Technology One Space Park Redondo Beach, CA 90278		 	
Revision/Change Record		Document Number	D39590
Revision	Document Date	Revision/Change Description	Pages Affected
B3	5-21-09	Tech Memo NP-EMD.2008.510.0069, Rev A. Update Generate cloud EDRs Code to be consistent with System Spec Rev N	1, 9, 10, 27
B4	8-13-09	Updated TBD01 (PCR019987)	Tables 19 & 32
B5	11-04-09	Updated for SDRL	All
B6	01-13-10	Implemented NP-EMD.2009.510.0048 Rev A VIIRS Geo Quality Flags Logic Updates	Table 14
B7	01-20-10	Implemented NP-EMD.2009.510.0072 Fix software bug in cloud layering in the Generate Cloud EDRs Algorithm, and prepared for TIM/ARB	2.2.2.14, 2.2.2.15
B	03-17-10	Incorporated comments from TIM and prepared for ACCB	All
C1	08-27-10	ECR1061/PCR024068 update output product ranges	2.2.1.2
C2	10-11-10	Updated due to document convergence including TM 2010.510.0013 and other administrative edits.	All
C3	09-21-11	Updated for PCR026486	21
C4	09-27-11	Updated OAD for PCR026627.	3, 7
C5	11-14-11	Updated for PCR027776	Table 7

Table of Contents

1.0 INTRODUCTION..... 1

 1.1 Objective..... 1

 1.2 Scope 1

 1.3 References 2

 1.3.1 Document References 2

 1.3.2 Source Code References 4

2.0 ALGORITHM OVERVIEW 5

 2.1 CCL Description 5

 2.1.1 Interfaces 6

 2.1.1.1 Inputs 7

 2.1.1.2 Outputs 8

 2.1.2 Algorithm Processing..... 8

 2.1.2.1 Retrieval Logic 9

 2.1.2.2 Configurable Parameters 11

 2.1.2.3 Aggregation Tables 11

 2.1.2.4 Main Module - cclDriver() 12

 2.1.2.5 setScan()..... 12

 2.1.2.6 ccl_cluster()..... 13

 2.1.2.7 ccl_product() 13

 2.1.2.8 cell_copy_out()..... 14

 2.1.2.9 recalcClusters() 14

 2.1.2.10 cloudDist() 14

 2.1.2.11 normDist () 14

 2.1.2.12 mbkm_first_guess() 15

 2.1.2.13 ekm_first_guess() 15

 2.1.2.14 apply_kmeans() 15

 2.1.2.15 assign_cloud_type () 15

 2.1.2.16 produceQualityFlags ()..... 16

 2.1.2.17 fillOutputs()..... 16

 2.1.2.18 loadAggTbl() 16

 2.1.2.19 getAggCell() 16

 2.1.2.20 assignScanLevelGeolocationData()..... 16

 2.1.3 Graceful Degradation..... 16

 2.1.4 Exception Handling 16

 2.1.5 Data Quality Monitoring 17

2.1.6 Computational Precision Requirements 17

2.1.7 Algorithm Support Considerations 17

2.1.8 Assumptions and Limitations 17

2.2 GCE Description..... 18

2.2.1 Interfaces 18

2.2.1.1 Inputs 19

2.2.1.2 Outputs 21

2.2.2 Algorithm Processing..... 22

2.2.2.1 Retrieval Logic 23

2.2.2.2 Main Module – GCE_main() 23

2.2.2.3 getScan() 24

2.2.2.4 average2d()..... 24

2.2.2.5 average3d()..... 24

2.2.2.6 Aggregate() 24

2.2.2.7 Height_Conversion() 24

2.2.2.8 posOutdata() 24

2.2.2.9 put_4_level() 24

2.2.2.10 put_dominant() 24

2.2.2.11 QF_Lyr_2d()..... 25

2.2.2.12 QF_Lyr_3d()..... 25

2.2.2.13 QF_2d()..... 25

2.2.2.14 sortLyrOrder() 25

2.2.2.15 sort_lyrs() 25

2.2.2.16 loadAggTbl() 25

2.2.2.17 getAggCell() 26

2.2.3 Graceful Degradation..... 26

2.2.4 Exception Handling..... 26

2.2.5 Data Quality Monitoring 26

2.2.5.1 Quality Flags 26

2.2.6 Computational Precision Requirements 27

2.2.7 Algorithm Support Considerations 27

2.2.8 Assumptions and Limitations 27

3.0 GLOSSARY/ACRONYM LIST 28

3.1 Glossary 28

3.2 Acronyms..... 31

4.0 OPEN ISSUES..... 33

List of Figures

Figure 1. CCL Algorithm Overview 6
 Figure 2. Cloud Cover Layers Overall Flow Diagram 7
 Figure 3. Conceptual Process for Cloud Cover Layers..... 9
 Figure 4. GCE Algorithm Overview 18
 Figure 5. Generate Cloud EDRs Overall Flow Diagram 19
 Figure 6. Conceptual Process for Generate Cloud EDRs..... 23

List of Tables

Table 1. Reference Documents 2
 Table 2. Source Code References..... 4
 Table 3. CCL Inputs 7
 Table 4. CCL Outputs 8
 Table 5. CCL/GCE Aggregation Tables..... 12
 Table 6. GCE Inputs 20
 Table 7. GCE Outputs..... 21
 Table 8. Glossary 28
 Table 9. Acronyms 31
 Table 10. TBXs 33

1.0 INTRODUCTION

1.1 Objective

The purpose of an Operational Algorithm Description (OAD) document is to express, in computer-science terms, the remote sensing algorithms that produce the Joint Polar Satellite System (JPSS) end-user data products. These products are individually known as Raw Data Records (RDRs), Temperature Data Records (TDRs), Sensor Data Records (SDRs) and Environmental Data Records (EDRs). In addition, any Intermediate Products (IPs) produced in the process are also described in the OAD.

The science basis of an algorithm is described in a corresponding Algorithm Theoretical Basis Document (ATBD). The OAD provides a software description of that science as implemented in the operational ground system.

The purpose of an OAD is two-fold:

1. Provide initial implementation design guidance to the operational software developer.
2. Capture the “as-built” operational implementation of the algorithm reflecting any changes needed to meet operational performance/design requirements.

An individual OAD document describes one or more algorithms used in the production of one or more data products. There is a general, but not strict, one-to-one correspondence between OAD and ATBD documents.

1.2 Scope

The scope of this document is limited to the description of the core operational algorithm(s) required to create the CCL related outputs:

VIIRS Cloud Layer Type IP
VIIRS Cloud Cover Type IP
VIIRS Cloud Aggregate Geolocation (GEO)

and the GCE related outputs:

VIIRS Cloud Base Height (CBH) EDR
VIIRS Cloud Cover/Layers (CCL) EDR
VIIRS Cloud Effective Particle Size (CEPS) EDR
VIIRS Cloud Optical Thickness (COT) EDR
VIIRS Cloud Top Height (CTH) EDR
VIIRS Cloud Top Pressure (CTP) EDR
VIIRS Cloud Top Temperature (CTT) EDR.

Note: GCE is referred to as Grid Cloud EDRs in some supporting documentation. The theoretical basis for these algorithms is described in the Cloud Cover/Layers Visible/Infrared Imager/Radiometer Suite ATBD.

The theoretical basis for the CCL and GCE algorithms are described in Sections 4.2 and 4.3, respectively, of the Cloud Cover/Layers Algorithm Theoretical Basis Document (ATBD), D0001-M01-S01-014.

1.3 References

1.3.1 Document References

The science and system engineering documents relevant to the algorithms described in this OAD are listed in Table 1.

Table 1. Reference Documents

Document Title	Document Number/Revision	Revision Date
Joint Polar Satellite System (JPSS) VIIRS Cloud Cover/Layers Algorithm Theoretical Basis Document (ATBD)	D0001-M01-S01-014	Latest
Joint Polar Satellite System (JPSS) Algorithm Specification Part 16	474-00448-01-16_JPSS-SRS-Vol-I-Part-16 474-00448-02-16_JPSS-DD-Vol-2-Part-16 474-00448-03-16_JPSS-OAD-Vol-III-Part-16 474-00448-04-16_JPSS-SRSPF-Vol-IV-Part-16	Latest
Operational Algorithm Description Document for the VIIRS Cloud Optical Properties (COP) Software	474-00074	Latest
Operational Algorithm Description Document for the VIIRS Cloud Top Parameters (CTP) EDR Software	474-00083	Latest
Operational Algorithm Description Document for the VIIRS Cloud Mask (VCM) Environmental Data Record (EDR) Software	474-00062	Latest
Operational Algorithm Description Document for VIIRS Geolocation (GEO) Sensor Data Record (SDR) and Calibration (Cal) SDR Software	474-00090	Latest
Joint Polar Satellite System (JPSS) Program Lexicon	470-00041	Latest
Operational Algorithm Description Document for the VIIRS Perform Parallax Correction (PPC) Software	474-00088	Latest
NGST/SE technical memo – Cross-granule Processing Memo	NP-EMD.2005.510.0038	7 Mar 2005
NGST/SE technical memo – VIIRS Cloud Mask (VCM) OAD Update	NP-EMD.2004.510.0050	3 Dec 2004
NGST/SE technical memo – MS Engineering Memo_CCL_GCE OAD Update	NP-EMD.2005.510.0077	7 Jul 2005
NGST/SE technical memo – NPP_VIIRS_GCE_QF Defns SPCR ALG980	NP-EMD.2006.510.0007	31 Jan 2006
NGST/SE technical memo – NPP_VIIRS_CCL_ARRAYS_INITIALIZATION	NP-EMD.2006.510.0057	7 Aug 2006
NGST/SE technical memo – NPP_VIIRS_GCE_LAYER_ORDERING_RevA	NP-EMD.2007.510.0034 Rev. A	8 Jun 2007
NGST/SE technical memo – NP-EMD-2008.510.0017_NPP_VIIRS_CCL_bug_fixes_and_new_cloud_mask	NP-EMD-2008.510.0017	26 Mar 2008
NGAS/AM&S technical memo – NPP_VIIRS_CCL_Cloud_Layer_Type_mismatch	NP-EMD.2008.510.0059	06 Nov 2008
NGAS/AM&S technical memo – Update Generate cloud EDRs to be consistent with System Spec Rev N	NP-EMD.2008.510.0069	4 Dec 2008

Document Title	Document Number/Revision	Revision Date
NGAS/SE technical memo – VIIRS Geo Quality Flags Logic Updates (PCR21471)	NP-EMD.2009.510.0048 Rev A	12 Oct 2009
NGAS/AM&S technical memo – VIIRS_NPP_GCE_bug_fix_in_layering (PCR22174)	NP-EMD.2009.510.0072	3 Dec 2009
NGAS/SE technical memo – PC_OAD_Last_Drop_Corrections	NP-EMD.2010.510.0013	22 Sep 2010
NGAS/SE technical memo – Modify CCL and GCE to correct an error in the aggregation of latitude and longitude	NP-EMD.2011.510.0004	11 Feb 2011
Joint Polar Satellite System (JPSS) Common Ground System (CGS) IDPS PRO Software User’s Manual Part 2	UG60917-IDP-1005	Latest
JPSS Algorithm Specification Volume II Data Dictionary for the Common Algorithms	474-00448-02-01	Latest
JPSS Algorithm Specification Volume II Data Dictionary for the VIIRS RDR/SDR- Block 2.0.0	474-00448-02-06	Latest

1.3.2 Source Code References

The science and operational code and associated documentation relevant to the algorithms described in this OAD are listed in Table 2.

Table 2. Source Code References

Reference Title	Reference Tag/Revision	Revision Date
VIIRS Cloud Cover/Layers – science-grade software	ISTN_VIIRS_NGST_3.5 Ver. 6 Rev. 3 (ECR A-057)	10 Aug 2005
VIIRS Cloud Cover/Layers – operational software	/PRO/EDR/VIIRS/clouds/ccl/ B1.5 Ver. D.1.1.6	01 Feb 2007
VIIRS Generate Cloud EDRs – science-grade software	ISTN_VIIRS_NGST_3.5 Ver. 6 Rev. 3 (ECR A-057)	10 Aug 2005
VIIRS Generate Cloud EDRs – operational software	/PRO/EDR/VIIRS/clouds/gce/ B1.5	Oct 2007
VIIRS Cloud Cover/Layers/ Generate Cloud EDRs – science-grade software	ISTN_VIIRS_NGST_3.5.1_DATA	21 Dec 2006
VIIRS Cloud Cover/Layers – science-grade software Includes TM: NP-EMD-2008.510.0017	ISTN_VIIRS_NGST_3.5.4_DATA (ECR A150 & A151)	16 Sep 2008
VIIRS Cloud Cover/Layers – operational software	B1.5x1 (OAD Rev B1)	26 Mar 2008
Modified GCE algorithm to re-implement TM NP-EMD.2007.510.0034.	(OAD Rev B2)	09 Nov 2008
VIIRS Cloud Cover/Layers – science-grade software, Includes TM: 2008.510.0059	ISTN_VIIRS_NGST_3.5.6_DATA	19 Nov 2008
VIIRS Generate Cloud EDRs – science-grade software	ISTN_VIIRS_NGST_3.5.7_DATA	29 Jan 2009
NGAS/AM&S technical memo – Update Generate cloud EDRs to be consistent with System Spec Rev N NP-EMD.2008.510. 0069.Rev-A	Build Post-X-I (OAD Rev B3)	26 May 2009
PCR019987	Sensor Characterization Build (SC-2)(OAD Rev B4)	13 Aug 2009
SDRL	OAD Rev B5	04 Nov 2009
Operational Software (Includes PCR21471)	Sensor Characterization (Build SC-6) (OAD Rev B6)	13 Jan 2010
Operational Software (Includes PCR22174)	Sensor Characterization (Build SC-6) (OAD Rev B7)	20 Jan 2010
ACCB (no code changes)	OAD Rev B	17 Mar 2010
ECR1061/PCR024068 update output product ranges	(OAD Rev C1)	27 Aug 2010
Convergence updates (No code updates)	(OAD Rev C2)	11 Oct 2010
PCR025519	Build 1.5.05.05 (OAD not updated)	01 Feb 2011
VIIRS Science Algorithms 3.5.10 (Includes TM 2011.510.0004)	ISTN_VIIRS_NGST_3.5.10_DATA	22 Feb 2011
PCR026486 (OAD Update to 3.5.10_Data) (x-ref PCR026485)	(OAD Rev C3)	21 Sep 2011
PCR026627 (ADL update to OAD)	(OAD Rev C4)	27 Sep 2011
PCR027776 (OAD updated from PCR027775-ISTN_VIIRS_NGST_4.26, ECR-ALG-0026)	(OAD Rev C5)	14 Nov 2011
OAD transitioned to JPSS Program – this table is no longer updated.		

2.0 ALGORITHM OVERVIEW

For this section, the definition of an algorithm is a logical grouping of operational algorithm modules for which there is a single Input-Processing-Output (I-P-O) architecture with a single defined set of external inputs and outputs (e.g., IPs or xDRs).

The retrieval of cloud cover and layers follows execution of the VIIRS CloudMask/Phase algorithm, COP algorithm, and CTP (Parameters) algorithm. These algorithms provide the necessary inputs describing the cloud mask, cloud phase, COP, and CTP (Parameters) for each pixel. The Perform Parallax Correction (PPC) module, which is also executed prior to the CCL, is assumed to produce parallax corrected IPs written in the identical format as the uncorrected IPs. CCL outputs are used as inputs to CBH and GCE modules.

The retrieval of generate cloud EDRs follows execution of the VIIRS CCL and CBH algorithms. The SDR MOD GEO, PPC IPs, CBH IPs, and CCL IPs and GEO are used as inputs to the GCE module. GCE produces EDRs which are end-products of the cloud processing chain and are not used by any other software units. GCE also produces Full resolution EDRs (FEDRs) which are expected to be used by other software units.

2.1 CCL Description

The CCL algorithm is implemented during IP/EDR processing and requires SDR, IP, and Look-Up Table (LUT) inputs to produce IP and GEO outputs. A top-level diagram for the CCL algorithm is shown in Figure 1.

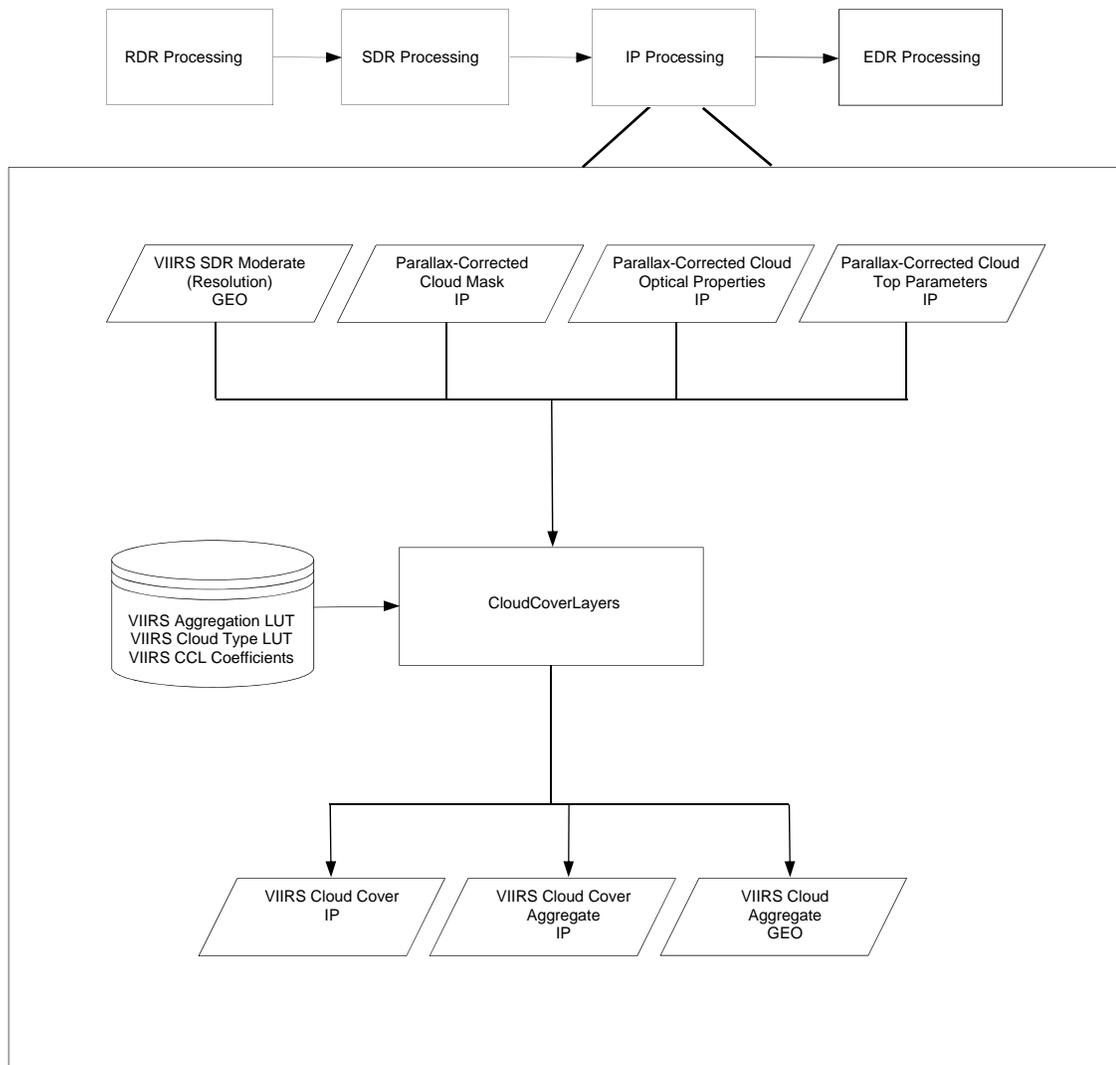


Figure 1. CCL Algorithm Overview

2.1.1 Interfaces

CCL consists of derived and core algorithm modules. The derived algorithm module ProEdrViirsCcl functions as a wrapper for the core algorithm module and handles the I-O stages in the I-P-O architecture. ProEdrViirsCcl initiates the core algorithm module Cloud Cover/Layers which makes up the P stage.

The main flow of the operational CCL algorithm is shown in Figure 2 below. The CCL algorithm gets all the required input data from DMS. When all the input data needed for processing is available, the core algorithm CloudCoverLayers is called to determine cloud cover layers. All inputs from and outputs to DMS are in binary format.

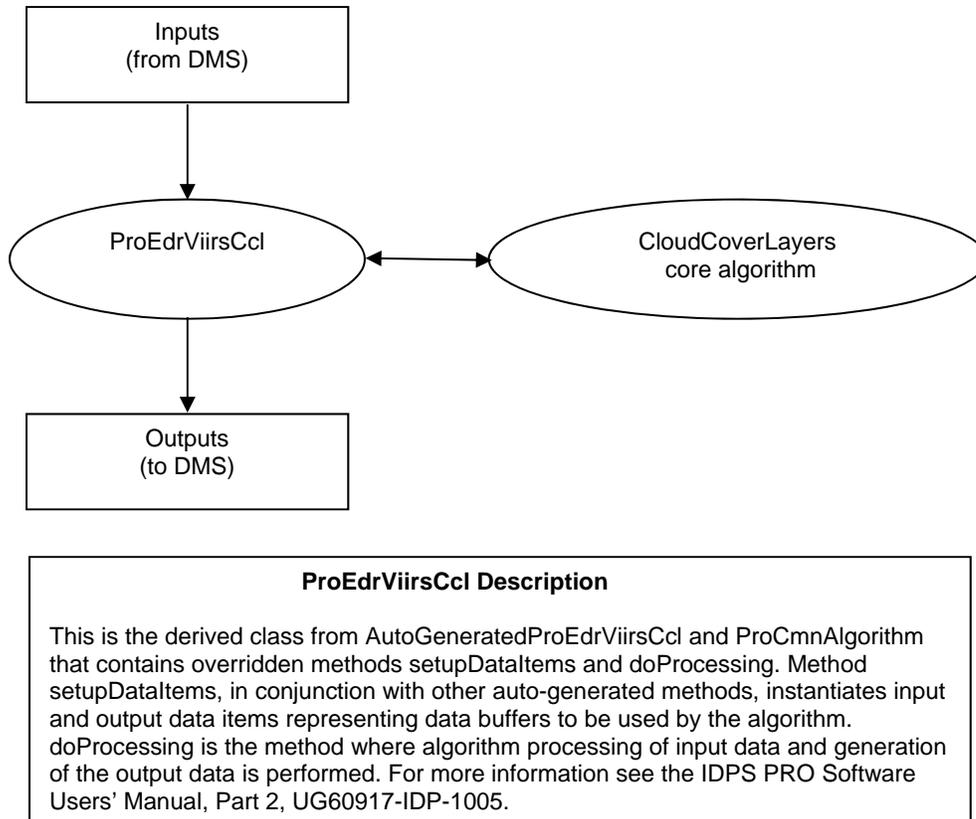


Figure 2. Cloud Cover Layers Overall Flow Diagram

The CCL Module requires VIIRS Moderate Geolocation (GEO), VIIRS Parallax Corrected Cloud Mask (CM) IP, VIIRS Parallax Corrected Cloud Optical Properties (COP) IP, VIIRS Parallax Corrected Cloud Top Parameters (CTP) IP, VIIRS Cloud Aggregation (AGG) LUT, VIIRS Cloud Type LUT, and VIIRS CCL IP Algorithm Coefficients as inputs to produce the following outputs: VIIRS Cloud Layer Type IP, VIIRS Cloud Cover Type IP, VIIRS Cloud Aggregate GEO. A detailed itemization of the inputs and outputs for the CCL Module is provided in Sections 2.1.1.1 and 2.1.1.2. See Section 2.3 of the Cloud Cover/Layers ATBD, D0001-M01-S01-014, for more information.

2.1.1.1 Inputs

CCL algorithm inputs are found in 474-00448-01-16_JPSS-SRS-Vol-I-Part-16, Table 3-1 (rows 6 - 8). Detailed descriptions of the inputs and LUTs are found in 474-00448-02-16_JPSS-DD-Vol-II-Part-16, and generalized below in Table 3.

Table 3. CCL Inputs

Input	Description	Reference Document
VIIRS Moderate Geolocation	Latitude, Longitude, Sensor and Satellite angles etc.	474-00448-02-06_JPSS-DD-Vol-II-Part-06
VIIRS Parallax Corrected Cloud Mask	Corrected Cloud Mask flags	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Parallax Corrected Cloud Optical Properties	Cloud Optical Thickness and Effective particle size etc.	474-00448-02-16_JPSS-DD-Vol-II-Part-16

Input	Description	Reference Document
VIIRS Parallax Corrected Cloud Top Parameters	Cloud Top Height, Cloud Top Temperature, Cloud Top Pressure etc.	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS CCL Coefficient	CCL tunable parameters	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Aggregate LUT	Cloud aggregated LUT values	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Type LUT	Cloud Type LUT values	474-00448-02-16_JPSS-DD-Vol-II-Part-16

2.1.1.2 Outputs

CCL algorithm outputs are found in 474-00448-01-16_JPSS-SRS-Vol-I-Part-16, Table 3-1 (4th row from end of table) and defined in 474-00448-02-16_JPSS-DD-Vol-II-Part-16. The CCL module writes three output products in binary format: VIIRS Cloud Layer Type IP; VIIRS Cloud Cover Type IP; VIIRS Cloud Aggregate GEO. The VIIRS Cloud Layer Type IP (see JPSS-DD-Vol-II-Part-16, Section 4.2) contains pixel level layer, type, and quality data. The VIIRS Cloud Cover Type IP (see JPSS-DD-Vol-II-Part-16, Section 4.3) contains total cloud cover, layer cloud cover, and layer cloud type for HCs. The VIIRS Cloud Aggregate GEO (see JPSS-DD-Vol-II-Part-16, Section 4.8) is produced separately from the IPs and is intended to be packaged with the output EDR products from GCE by Data Delivery.

CCL algorithm outputs are found in 474-00448-01-16_JPSS-SRS-Vol-I-Part-16, Table 3-1 (4th row from end of table) and defined in 474-00448-02-16_JPSS-DD-Vol-II-Part-16. The CCL module writes three output products in binary format: VIIRS Cloud Layer Type IP; VIIRS Cloud Cover Type IP; VIIRS Cloud Aggregate GEO. The VIIRS Cloud Layer Type IP (see JPSS-DD-Vol-II-Part-16, Section 4.2 Table 12) contains pixel level layer, type, and quality data. The VIIRS Cloud Cover Type IP (see JPSS-DD-Vol-II-Part-16, Section 4.3 Table 13) contains total cloud cover, layer cloud cover, and layer cloud type for HCs. The VIIRS Cloud Aggregate GEO (see JPSS-DD-Vol-II-Part-16, Section 4.8 Table 14) is produced separately from the IPs and is intended to be packaged with the output EDR products from GCE by Data Delivery.

Table 4. CCL Outputs

Output	Description	Reference Document
The VIIRS Cloud Layer Type IP	VIIRS-Cd-Layer-Type-IP contains pixel level layer, type, and quality data	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Cover Type IP	VIIRS-Cd-Cov-Type-IP contains total cloud cover, layer cloud cover, and layer cloud type for HCs	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Aggregate GEO	VIIRS-CLD-AGG-GEO contains the aggregate Geolocation values	474-00448-02-16_JPSS-DD-Vol-II-Part-16

2.1.2 Algorithm Processing

The purpose of the CCL module is to classify cloudy pixels into as many as four cloud layers and to determine the cloud type for each layer. The module produces Intermediate Products (IPs) that describe the pixel-level assignment of the cloud layers and types plus cloud cover

(CC) and cloud type (CT) products generated over horizontal cells (HC) and reported for each of the four layers plus averaged over all layers. See Section 2.5 of the Cloud Cover/Layers ATBD, D0001-M01-S01-014, for more information. The dataflow model for the CCL algorithm is illustrated in Figure 3.

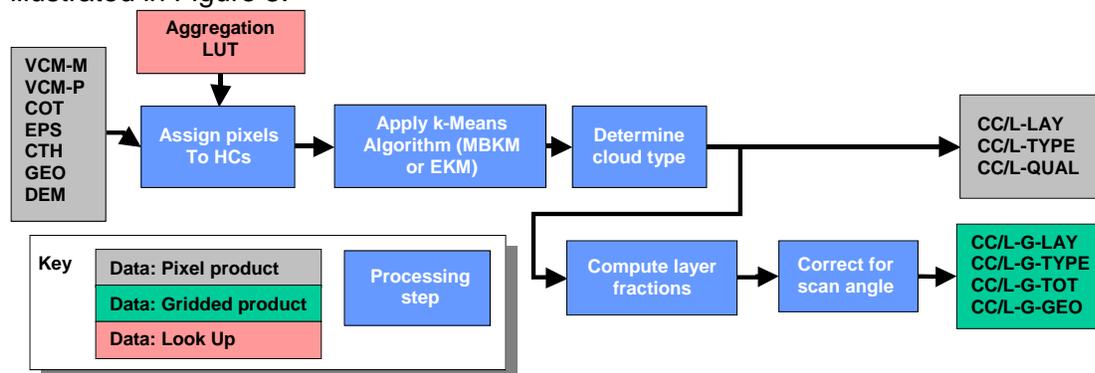


Figure 3. Conceptual Process for Cloud Cover Layers

The CCL reads and processes the data one scan at a time. Horizontal product cells are always comprised of pixels within a single scan. The assignment of pixels to product cells is accomplished using an aggregation table. For each scan line, CCL loops over all product cells and generates outputs based on the pixel-level cloud IPs identified with each cell. The aggregation approach developed for the CCL allows for a larger group of pixels surrounding each HC (or product cell) to better identify the layers via the k-means analysis. This extended region is referred to as the clustering cell. It includes pixels from the adjacent scans. The first and last scan lines in a granule require that the cluster cells contain pixel information from the previous/next scans of the preceding and succeeding granules. To accomplish this, the last scan of the previous granule and the first scan of the next granule is read and passed to the algorithm as the cluster cells when the product cell is the first or last scan. CCL uses a method called setScan() to read the previous, current and next scan input data. The aggregation table is read using loadAggTbl() and the pixels associated with each cell are identified with getAggCell(). The CCL routine keeps data from the previous, current, and next scans in memory as the information from all three scans may be required in the clustering analysis. See Section 2.4 of the Cloud Cover/Layers ATBD, D0001-M01-S01-014, for more information.

Note: It should be noted that reading one scan before and one scan after the current granule differs somewhat from the science code that reads two scans from the previous granule and nothing from the next granule. Using the method defined in the science code would have shifted the output by one scan, therefore the one scan before and the one scan after method was used to avoid shifting the output.

2.1.2.1 Retrieval Logic

The CCL module uses a preprocessing routine called cell_copy_in() which copies the cloud IP data associated with each cell to temporary arrays. Each pixel in the cell is identified as either 1=product, 2=cluster, or 3=product and cluster pixels. In practice, all pixels in the product cell are also used in the clustering analysis so pixels are identified as 1 or 3 in normal operating modes. Pixels assigned to each cell may originate from the previous, current, or next scans; however product pixels are always associated with the current scan. The preprocessing routine also computes the mean latitude, longitude, and sensor zenith angle for the product cell. Logic

is included in `cell_copy_in()` to handle missing data. Two options for dealing with missing data are available as configuration parameters. Option 1 sets the k-means cluster weights for COT and EPS to zero if values for these inputs are missing. If CTH is missing, then the pixel is not used in the clustering calculations. Option 2 eliminates any pixel that is missing CTH, COT, or EPS from the k-means clustering evaluation. The k-means weight defaults to the values specified in the configuration file if the input CTH, COT, and EPS are available. Cloud phase is converted to a float with 0.0 indicating water, 0.5 indicating mixed phase, partly cloudy or overlapping cloud, and 1.0 indicating cirrus or opaque ice phase. Any pixels not identified as confident cloudy in the VIIRS cloud mask are treated as clear by the CCL algorithm.

Pixels identified with the overlapping cloud phase are a problem for the CCL routine as these pixels represent information from more than one layer. This problem is addressed by excluding the multi-layer pixels from the process of layer determination in one of two ways:

- (1) Multi-layer pixels assigned as “cluster” cell only are omitted from the current “product” cell analysis.
- (2) Multi-layer pixels assigned as “product-plus-cluster” cell are reassigned as “product” cell only for the current “product” cell analysis. Therefore, the pixels are included in the cloud fraction determination for the horizontal cell but omitted from the layer/type statistic analysis.

The CCL algorithm is divided into two parts. The first, `ccl_cluster()`, performs the analysis of cloud layer based on the selected inputs and produces pixel-level assignment of cloud layer and type for the cell. If no pixels in a given cell are cloudy then the `ccl_cluster()` function returns fill values for the cloud layer and type for the cell.

`Ccl_cluster` function determines a “pixel-level” cloud layer and cloud type. This is an IP output, not EDR. The cloud fraction is determined from the number of cloudy pixels in the HC and is independent of the cloud layer and type determined in the

`CCL_Cluster` function. If there is no cloud in the product cell, i.e. the horizontal cell, the cloud fraction is zero, not FILL. The product quality is computed based on the parameters used in the k-mean analysis.

The first guess module makes the initial assignment of cloudy pixels into layers. Two algorithms are available. The function `mbkm_first_guess()` executes the modified baseline routine and assigns the initial cloud layers based on CTH thresholds (at 2.5, 5.0 and 7.5 km). The function `ekm_first_guess()` executes the extended algorithm and assigns initial cloud layers based on a statistical analysis of CTH. This routine begins with all cloudy pixels in one cluster. If the standard deviation of the cloud heights is greater than the specified threshold (0.75 km), then an attempt is made to split the clouds into two layers. This is accomplished via a k-means distance clustering using height as the only variable. The standard deviations of the resulting cluster are compared with the separation in mean height between the clusters. If this quantity exceeds a specified threshold (1.6 km) or the standard deviation of the original cluster exceeds a threshold (1.6 km), then the cloudy pixels are split into the two layers. The process repeats for the layer with the largest standard deviation in CTH until a maximum of four layers is defined.

Once the initial layer assignments are complete, the `apply_kmeans()` routine is used to refine the layer assignments based on the variables (i.e., CTH, COT, EPS, and phase) specified in the configuration file. The k-means algorithm compares normalized distance relative to the mean for each layer based on all selected input variables. Cloudy pixels are re-assigned to layers based on the minimum distance. The algorithm iterates a number of times,

KMEANS_MAXITER, and updates the layer means with each iteration. If the number of pixel reassignments is less than a critical value, CLUSTER_CONVERG_CRIT, then the process is terminated.

Finally the CTs are assigned for each pixel in the function assign_cloud_type(). For each pixel, the cloud phase determines the possible CTs. The CTH, COT, and EPS of the pixel are compared with the mean values for each CT and the minimum distance is used as the criteria for assigning the type.

The second component of the CCL algorithm, ccl_product(), processes the pixels in the product cell to derive the CC fraction and CT for each cloud layer. This component includes the oblique angle correction to the CC fraction estimates. In a given layer, it is possible for cloudy pixels within an HC to be classified as different types (e.g., if the phase is different). If this situation should arise, the code assigns the prevalent CT to the HC. The ccl_product() code also includes a preprocessing step used to re-number the cloud layers assigned by the EKM algorithm if the product cell contains fewer layers than the cluster cell. At the end of ccl_product(), the aggregation table data is used to assign pixel-level values from each cell to the appropriate scan locations.

2.1.2.2 Configurable Parameters

Configurable inputs to the CCL algorithm are specified in a single configuration file. The configurable parameters are summarized in JPSS-DD-Vol-II-Part-16, Section 7.2.2.2.2-1. The table identifies the mean values of CTH, COT, and EPS for each of five cloud types. A set of normalization factors (or weights) are used to specify the relative importance of CTH, COT, EPS, and cloud phase in the k-means clustering analysis. Use of these parameters can be turned off by setting the values to zero. Non-zero factors represent the range of variability of each parameter. Careful selection of these parameters provides the desired balance between the variables in the k-means algorithm.

Two options for dealing with missing data are available in the code and can be selected with the GRACEFUL_DEGRADATION configuration parameter. If COT and/or EPS are selected as k-means parameters (i.e., non-zero normalization) and are not present for a given pixel, then either the pixel is omitted from the analysis (IGNORE_PIXEL) or the missing variable is omitted from the analysis (IGNORE_VARIABLE). If CTH is missing, then the pixel is omitted from the analysis because CTH is required by the first guess algorithm. Cloud phase is always required for the cloud layer analysis.

The modified baseline k-means (MBKM) algorithm is based on an initial guess that groups cloudy pixels into layers based on fixed height thresholds (at 2.5, 5.0, and 7.5 km). The extended k-means (EKM) algorithm determines the initial estimate of the cloud layers based on a statistical analysis of the cloud height distributions. These options can be selected with the KMEANS_ALGORITHM configuration parameter.

For each cloud type the mean values of CTH, EPS, and COT are provided. The types appropriate for each cloud phase (i.e., water, ice and mixed) are determined by limits set in the assign_cloud_type() function (e.g., water = type 0 to 2, ice = type 1 to 4, mixed = type 1 to 2).

2.1.2.3 Aggregation Tables

The product cell size for the VIIRS Cloud aggregated output is approximately 6 km from nadir to end-of-scan (EOS). The product cells are constructed from pixels in one half of the 16 pixel

scan. At nadir, the product cell size is 8 by 8 pixels. At EOS, the product cell is 4 by 4 pixels. In total there are 2 x 508 product cells per scan. The cluster cell size defines the region surrounding the product cell that is used to identify and classify the cloud layers. Three cluster cell sizes are considered: 6 km, 12 km, and 18 km (currently, only the aggregation table with the 12 km cluster cell size is being used). Aggregation tables are used to identify the pixels belonging to each product/cluster cell. Table 5 summarizes aggregation tables that have been generated for VIIRS data.

Table 5. CCL/GCE Aggregation Tables

Name	Lines per Scan	SPA	Product Cell Size	# of Cells	Nadir Pixels	EOS Pixels	Cluster Cell Size		
viirs06	16	Yes	6 km	508x2	8x8	4x4	6 km	12 km	18 km

2.1.2.4 Main Module - cclDriver()

This function is the main driver for the core CCL algorithm. Called by ProEdrViirsCcl(), cclDriver() sets the scan pointers, loads the aggregation table data, prefills output data buffers, sets (produces) output quality flags, determines the total number of cells per scan, and processes valid scans in the granule. For each cell in a scan, cell_copy_in() copies cell data into temporary variables for calculations, ccl_cluster() calculates pixel-level cloud layers and types, ccl_product() calculates aggregate-level cloud layers and types, and cell_copy_out() writes pixel-level cloud layer and type to the output.

2.1.2.5 setScan()

cclDriver() calls setScan() to set the scan pointers (previous, current, next) used in processing to the appropriate locations in the IP data.

cell_copy_in() copies in the pixel-values for all input variables as identified by the aggregation table for a given cell.

- getAggCell() is called to retrieve data from the aggregation table for the current cell.
- Sensor zenith angles, latitudes and longitudes are computed for the product cell (for previous, current, and next scans). Latitudes and longitudes are computed as follows:

$$\begin{aligned}
 x &= \sin(\text{co_Lat}) * \cos(\text{Lon}) \\
 y &= \sin(\text{co_Lat}) * \sin(\text{Lon}) \\
 z &= \cos(\text{co_Lat})
 \end{aligned}$$

where co_Lat = 90-Lat

- COT, EPS, CTH, Phase, and Cloud Confidence data is obtained for cluster defined pixels in a cell for previous, current, and next scans.
- For confidently cloudy pixels with valid cloud phase, if the pixels are missing COT, or EPS, the pixels are ignored (if mode set to IGNORE_PIXEL) or the corresponding pixel COT and EPS weights are set to zero (if mode set to IGNORE_VARIABLE).
- For confidently cloudy pixels with valid data, the phase is assigned according to cloud mask.
- The clustering flag is set based on percentage of cloudy product pixels in the cell.

- Aggregate solar zenith, solar azimuth, sensor zenith, sensor azimuth angles and latitude and longitude are calculated for the cell. The solar and sensor angles are averaged, while latitudes and longitudes are computed as follows:

```
Calculate average for each value: x, y, z
sqrt_result = sqrt( xavg2+yavg2+zavg2)
cell latitude = acos(zavg / sqrt_result)
cell longitude = atan2(yavg, xavg)
```

2.1.2.6 ccl_cluster()

The purpose of `ccl_cluster()` is to apply the k-means algorithm to all pixels in the cluster cell and to set the pixel-level layer and type values. A first guess at layer (also referred to as cluster) assignments for pixels is used to initialize the k-means cluster analysis. The configurable parameter `kmeansAlgorithm` selects which of the two available first guess algorithms to use (`mbkm_first_guess()` or `ekm_first_guess()`). The default is `mbkm_first_guess()`. At this point, assignments of layers to pixels were based on CTH thresholds (if `mbkm` algorithm was used) or CTH statistical analysis (if `ekm` algorithm was used).

Cluster results from the first guess algorithm are used by `apply_kmeans()` to further refine the layer assignments by reassigning pixels to layers using CTH, COT, EPS, and PHASE (involves k-means calculations).

The output IP pixel layer values are set using cluster results from `apply_kmeans()`.

`Assign_cloud_type()` processes cluster means and cluster count results from `apply_kmeans()` to set the output IP pixel type values.

2.1.2.7 ccl_product()

`ccl_product()` determines aggregate layer type and cloud cover layer (fraction) based on product cell pixel-level layer and type assignments.

If the EKM algorithm was used, layer adjustments are computed for layers that do not contain any product pixels. Beginning with the lowest layer that does not contain product pixels, all product pixels assigned to layers higher than the empty layer are shifted down one layer.

For all product pixels in a cell that contain valid layer and type data, cloud and cluster counters are incremented and the CTHs for each layer are accumulated in `cluster_cth_means`. Next, the cloud type with the highest occurrence in a layer is assigned as the cloud type for that layer in the cell.

Cloud fraction for the cell is then calculated as follows:

- Determine apparent fractional coverage.
 $\text{Appar_frac} = \text{cloud count} / \text{product pixels}$
- Use the apparent fractional coverage to determine cloud fraction from the cloud fraction look-up table.
- Determine horizontal cloud top height (i.e. cloud top height for the cell).
 $\text{Hcth} = \text{cluster count} * \text{cluster means}$

4. Use the horizontal cloud top height to determine cloud altitude from the cloud altitude look-up table.
5. Determine cloud fraction view angle correction parameters.
Use the cloud fraction and cloud altitude as indexes into the gamma look-up table to obtain the correct cloud masking exponent (gamma).

$$\theta = \text{sensor zenith angle for the cell}$$

$$x_value = 1.0 + \theta * \tan(\theta) + (1./\cos(\theta))$$

$$x_value = 2.0 / x_value$$

$$f_value = \text{pow}(x_value, \text{gamma})$$
6. Compute cloud fraction.

$$\text{Cloud frac} = f_value * \text{cluster count} / \text{product pixels}$$

Refer to Cloud Cover/Layers ATBD in the Scan Angle Effect and Correction for Cloud Fraction section.

2.1.2.8 cell_copy_out()

cell_copy_out() is called by cclDriver() to write pixel cloud layer and type for only current scan product pixels to the output IP.

2.1.2.9 recalcClusters()

recalcClusters() is called by apply_kmeans() to calculate cluster counts, cluster means and phase counts for cloud layers. This function only processes pixels defined as cluster pixels. The frequency of each cloud phase (water, ice, mixed) for cluster pixels in a cell is determined for later use by assign_cloudType(). Also, the CTH, COT, EPS and Phase values are summed up for applicable layers. Counters and checksums are used to ensure that the correct number of pixels was processed for the cell. CTH, COT, EPS and Phase means are calculated.

2.1.2.10 cloudDist()

Called by assign_cloudType(), this function measures the normalized “distance” between cluster means (if valid) and Cloud Type Table values (COT, CTH, EPS) for the cloud type indicated. The following formula is used:

$$\begin{aligned} & (\text{COT cluster means} / \text{cloud type table COT})^2 \\ & + (\text{CTH cluster means} / \text{cloud type table CTH})^2 \\ & + (\text{EPS cluster means} / \text{cloud type table EPS})^2 \\ & = \text{normalized distance} \end{aligned}$$

2.1.2.11 normDist ()

This function normalizes the “distance” between pixel values and cluster means for CTH, COT, EPS, and Phase (where valid) for different cloud phases using weighted coefficients. Note: if the graceful degradation mode is set to IGNORE_VARIABLE, the weights for COT and EPS are zero. The following formula is used:

$$\begin{aligned} & ((\text{COT} - \text{COT cluster means}) / \text{COT weight})^2 \\ & + ((\text{CTH} - \text{CTH cluster means}) / \text{CTH weight})^2 \\ & + ((\text{EPS} - \text{EPS cluster means}) / \text{EPS weight})^2 \\ & + ((\text{Phase} - \text{Phase cluster means}) / \text{Phase weight})^2 \\ & = \text{normalized distance} \end{aligned}$$

2.1.2.12 mbkm_first_guess()

The purpose of mbkm_first_guess() is to provide the first guess at layer identification based on fixed cloud top height ranges. For each product or product-and-cluster pixel in a cell that contains valid phase data, the CTH is compared to the cloud top height thresholds (low, mid, high) starting with low. CTH thresholds correspond to cloud layers as follows:

Layer 0: CTH <= CTH_LOW_THRESH
Layer 1: CTH <= CTH_MID_THRESH
Layer 2: CTH <= CTH_HIGH_THRESH
Layer 3: CTH > CTH_HIGH_THRESH

Cluster counters keep track of the number of pixels assigned to each layer, including the NULLAYER used to capture pixels that are PRODUCT only. Cluster counts for each layer are used as checksums to verify all pixels in the cell were processed.

2.1.2.13 ekm_first_guess()

The EKM algorithm begins with all cloudy pixels assigned to a single layer. If the standard deviation of CTH is greater than some threshold (e.g., 0.75 km) then an attempt is made to assign the cloudy pixels into two layers. This is accomplished via a Euclidean distance metric with CTH as the only variable. Once the clusters are established, the standard deviation of CTH for each distribution is computed, as is the separation between the mean values of CTH. The clusters are accepted as distinct if the separation divided by the sum of the two standard deviations is greater than a prescribed threshold value (e.g., 1.5). Alternatively, if this is not the case but the standard deviation of the initial combined group is greater than yet another critical value (e.g., 1.6 km), then the two clusters are also retained. This process is repeated for the cluster with the largest standard deviation in CTH until a maximum of four cloud layers is established. The three thresholds (EkmCthThresh1, EkmCthThresh2, EkmCthThresh3) used by this algorithm are tunable parameters.

2.1.2.14 apply_kmeans()

The apply_kmeans() algorithm calls recalClusters() to initially calculate the cluster means.

Next, this function loops through the following steps (get normalized distance (call normDist()), set cluster pixel to minimum distance, calculate cluster means (call recalClusters()), test for convergence) to iteratively reassign pixels to layers until cluster convergence criteria is met or the maximum number of iterations is reached.

2.1.2.15 assign_cloud_type ()

In ccl_cluster(), assign_cloud_type() is called to determine the frequency of each cloud phase (water, ice, mixed) for the pixels in a cell and to assign the phase with the highest frequency to the cell. The cell phase is then compared to phase constants to obtain the max/min indexes for the range of cloud types associated with that phase in the cloud type table. Cluster means of CTH, COT, and EPS for the cell are compared to cloud types within range in the cloud type table using cloudDist(). The result with the minimum distance is assigned as the cloud type.

2.1.2.16 produceQualityFlags ()

In cclDriver() initialization, produceQualityFlags() is called to set the output quality flags for every pixel in the granule according to cloud mask flags for cloud confidence, surface type, sun glint, and snow/ice. This function requires pixel locations, cloud mask data, and location of quality flags to be filled.

2.1.2.17 fillOutputs()

During CCL cell-level processing, if a cell-level function fails, then fillOutputs() is called to fill the CCL outputs with error fill values and the quality flags with zero for the indicated cell and all pixels associated with that cell. This function requires pixel and cell locations to be filled.

2.1.2.18 loadAggTbl()

Used by CCL and GCE algorithms, the loadAggTbl() function takes each entry in the aggregation table (an entry contains pixel row/col, scan, cell row/col, product) and populates a temporary aggregation structure containing an array of cells with corresponding data for those cells. As the cells are populated with data, a counter logs and stores the number of pixels that were mapped to a cell.

2.1.2.19 getAggCell()

Used by CCL and GCE algorithms, the getAggCell() function accesses the temporary aggregation structure to get data for the indicated cell. Cell data includes: pixel indexes; pixel scan values (previous, current, next); pixel product masks (product, cluster, product and cluster); total number of pixels mapped to the cell.

2.1.2.20 assignScanLevelGeolocationData()

After looping through the scans in cclDriver(), this function is called with the current cross-granule data and the geolocation output pointers as its parameters. Its basic function is simply to copy scan-level fields from the VIIRS-MOD-GEO input over to the VIIRS-CLD-AGG-GEO output.

2.1.3 Graceful Degradation

The CCL module does not implement graceful degradation. CCL retrievals rely entirely on the availability of data from the VIIRS cloud IPs. No logic is included to replace missing data with secondary sources.

2.1.4 Exception Handling

Error-handling in the Input (I) and Output (O) stages of the I-P-O algorithm addresses errors associated with reading/writing of databases. If an error occurs, the error is reported, the process is terminated and no outputs are produced.

Error-handling in the Processing (P) stage involves granule level, cell level, and pixel level errors. All outputs are initialized with Not Applicable (NA) FILL (except quality flags which are initialized to zero) so all unprocessed cell level or pixel level data contains FILL values.

For granule level errors, i.e. errors resulting from loading the aggregation table or an invalid number of scans, the error is reported, the process is terminated and no outputs are produced.

For pixel level errors, the errors are reported and the pixel level function fails back to the cell level function call.

For cell level errors, the errors are reported, cell level outputs and pixel level outputs associated with the cell being processed are filled with Error (ERR) FILL (quality flags are set to zero) and processing continues with the next cell.

2.1.5 Data Quality Monitoring

The CCL algorithm produces IP and GEO outputs, therefore quality assessment and diagnostics are not done in this algorithm. Any Data Quality Notification (DQN) produced for the Cloud algorithms are handled by the Generate Clouds EDR (GCE) process.

2.1.6 Computational Precision Requirements

The CCL module employs single precision throughout.

2.1.7 Algorithm Support Considerations

Aggregation tables used by the CCL and GCE modules are based on the current VIIRS sensor design. Changes to the design (e.g., sub-pixel aggregation boundaries, number of pixels per line) may require an off-line recalculation of the aggregation tables.

2.1.8 Assumptions and Limitations

The CCL algorithm assumes that the VIIRS 750m SDR auxiliary data, the VCM EDR including Cloud Phase, the COP IP, and CTP (Parameters) IP are all available for processing. All inputs are expected in binary format at M-band pixel resolution.

2.2 GCE Description

The GCE algorithm is implemented during EDR processing and requires GEO, IP, and Look-Up Table (LUT) inputs to produce the cloud EDR outputs averaged over HCs and report the sum over all layers for each cloud layer. A top-level diagram for the GCE algorithm is shown in Figure 4. See Section 3.2 of the Cloud Cover/Layers ATBD, D0001-M01-S01-014, for more information.

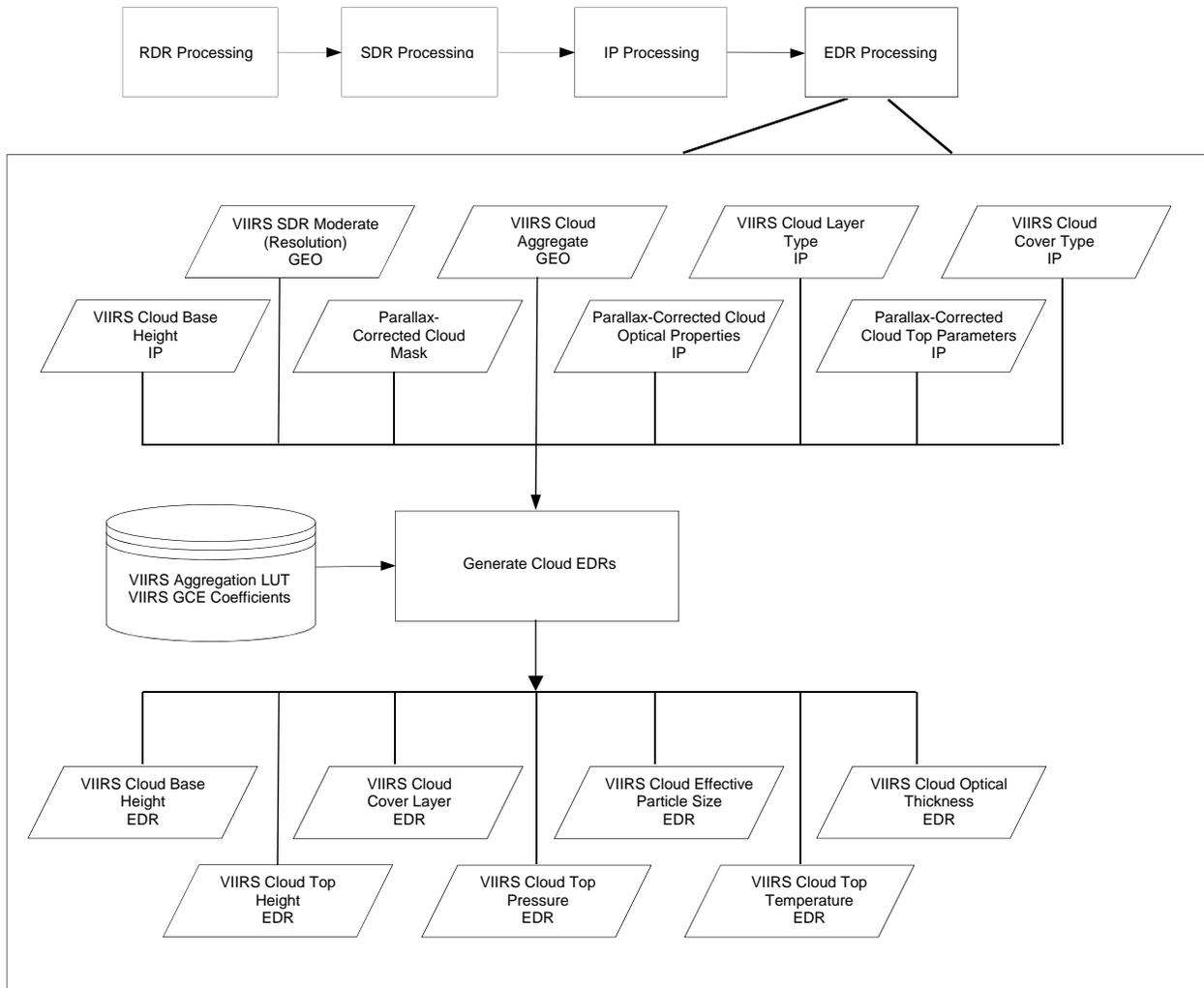


Figure 4. GCE Algorithm Overview

2.2.1 Interfaces

GCE consists of derived and core algorithm modules. The derived algorithm module ProEdrViirsGce functions as a wrapper for the core algorithm module and handles the I-O stages in the I-P-O architecture. ProEdrViirsGce initiates the core algorithm module GenerateCloudEDRs which makes up the P stage.

The main flow of the operational GCE algorithm is shown in Figure 5 below. The GCE algorithm gets all the required input data from DMS. When all the input data needed for processing is

available, the core algorithm GenerateCloudEDRs is called to produce cloud EDRs. All inputs from and outputs to DMS are in binary format.

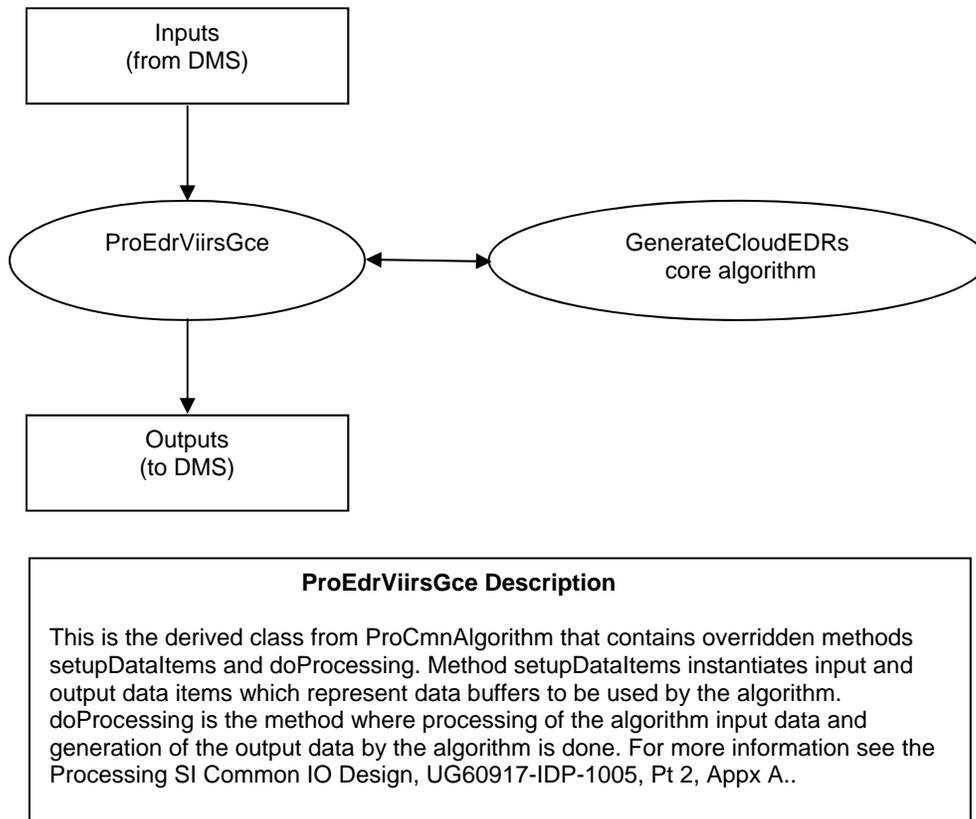


Figure 5. Generate Cloud EDRs Overall Flow Diagram

The GCE Module requires VIIRS Moderate Geolocation (GEO), VIIRS Cloud Aggregate GEO, VIIRS Cloud Base Height (CBH) IP, VIIRS Cloud Cover Type IP, VIIRS Cloud Layer Type IP, VIIRS Parallax Corrected Cloud Optical Properties (COP) IP, VIIRS Parallax Corrected Cloud Top Parameters (CTP) IP, VIIRS Cloud Aggregation (AGG) LUT, and VIIRS GCE IP Algorithm Coefficients as inputs to produce the following outputs: VIIRS Cloud Base Height EDR, VIIRS Cloud Cover Layer EDR, VIIRS Cloud Effective Particle Size EDR, VIIRS Cloud Optical Properties EDR, VIIRS Cloud Top Height EDR, VIIRS Cloud Top Pressure EDR, and VIIRS Cloud Top Temperature EDR. See Section 2.3 of the Cloud Cover/Layers ATBD, D0001-M01-S01-014, for more information.

VIIRS Parallax Corrected Cloud Mask (CM) IP is an optional input used only for metadata purposes. A detailed itemization of the inputs and outputs for the GCE Module is provided in Sections 2.2.1.1 and 2.2.1.2.

2.2.1.1 Inputs

GCE algorithm inputs are found in 474-00448-01-16_JPSS-SRS-Vol-I-Part-16, Table 3-1 (rows 19-23 of table) and defined in 474-00448-02-16_JPSS-DD-Vol-II-Part-16 (see below Table 6 for reference). No range checking of these inputs is performed in the code. The data is expected

to be non-scaled. Any scaled data is un-scaled by the Processing Common Code prior to using it in this algorithm.

Table 6. GCE Inputs

Input	Description	Reference Document
VIIRS Moderate Geolocation	Latitude, Longitude, Sensor and Satellite angles etc.	474-00448-02-06_JPSS-DD-Vol-II-Part-06
VIIRS Parallax Corrected Cloud Mask	Corrected Cloud Mask flags	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Parallax Corrected Cloud Optical Properties	Cloud Optical Thickness and Effective particle size etc.	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Parallax Corrected Cloud Top Parameters	Cloud Top Height, Cloud Top Temperature, Cloud Top Pressure etc.	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS GCE Algorithm Coefficient	GCE tunable parameters	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Aggregate GEO	Cloud Aggregated Geo values	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Aggregate LUT	Cloud aggregated LUT values	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Base Height IP	Cloud Base Height (CBH) data	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Layer Type	Cloud Layer data	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Cover Type	Cloud Cover data	474-00448-02-16_JPSS-DD-Vol-II-Part-16
Data Quality Threshold Table for CBH	VIIRS CBH DQTT contains threshold used to trigger the DQN as well as the text contained in the DQN for CBH	474-00448-02-16_JPSS-DD-Vol-II-Part-16
Data Quality Threshold Table for CCL	VIIRS CCL DQTT contains threshold used to trigger the DQN as well as the text contained in the DQN for CCL	474-00448-02-16_JPSS-DD-Vol-II-Part-16
Data Quality Threshold Table for CEPS	VIIRS CEPS DQTT contains threshold used to trigger the DQN as well as the text contained in the DQN for CEPS	474-00448-02-16_JPSS-DD-Vol-II-Part-16
Data Quality Threshold Table for COT	VIIRS COT DQTT contains threshold used to trigger the DQN as well as the text contained in the DQN for COT	474-00448-02-16_JPSS-DD-Vol-II-Part-16
Data Quality Threshold Table for CTH	VIIRS CTH DQTT contains threshold used to trigger the DQN as well as the text contained in the DQN for CTH	474-00448-02-16_JPSS-DD-Vol-II-Part-16
Data Quality Threshold Table for CTP	VIIRS CTP DQTT contains threshold used to trigger the DQN as well as the text contained in the DQN for CTP	474-00448-02-16_JPSS-DD-Vol-II-Part-16
Data Quality Threshold Table for CTT	VIIRS CTT DQTT contains threshold used to trigger the DQN as well as the text contained in the DQN for CTT	474-00448-02-16_JPSS-DD-Vol-II-Part-16

2.2.1.2 Outputs

CBH algorithm outputs (and Quality Flags) are found in 474-00448-01-16_JPSS-SRS-Vol-I-Part-16, Table 3-1 (5th row from end of table) and defined in 474-00448-02-16_JPSS-DD-Vol-II-Part-16, Section 5.

EDRs include:

- CBH,
- CCL,
- CEPS,
- COT,
- CTH,
- CTP (Pressure),
- CTT,

The GCE module generates cloud EDRs reported over HCs. A common HC size is used for all cloud EDRs (i.e., approximately 6 km). These products are in binary format. Along with the EDR output, QF values are also reported over the HC. Both scaled (integerized) and full resolution (nonscaled) EDRs are produced (refer to IDPS Scaled Integers Proposed Solution). Scaled EDRs include CBH, CCL, CEPS, COT, CTH, CTP (Pressure), and CTT. Full resolution EDRs (FEDRs) include CCL, CTH, and CTP (Pressure).

Each EDR contains two types of data (layered and total) and three types of quality flags (layered, total, and non-cloud related). CCL EDR also contains type data. Layered data contains the retrieved values of, for example, COT on a layer-by-layer basis, and total contains the average over the HC (except for cloud cover, which represents the total fraction over the HC). Additionally, scaled EDRs contain scale and offset values whereas full EDRs do not. Furthermore, layered data are arranged according to the layered CTH, where the 1st layer has the highest cloud height, and the 2nd layer has the next lower cloud height, etc. As such, no cloud will ever appear in layer 2 or 3 when the 1st layer is FILL.

In the “layered” and “total” COT, EPS, CTH, CTT, CTP and CBH the data field will be ‘FILL’ if there is no cloud in the designated layer or HC. For layered and “total” cloud cover, however, the data field is set to “0” if there is no cloud in the designated layer or HC. For cloud type, the type will be “FILL” if there is no cloud in the designated layer.

Quality flags are described in detail in all the cloud EDRs, see JPSS-DD-Vol-II-Part-16, Section 5. These quality flags have essentially the same meaning in each EDR except for those QFs where specific thresholds are used (for example, COT and EPS have different out-of-bound threshold values).

Table 7. GCE Outputs

Output	Description	Reference Document
VIIRS Cloud Base Height	VIIRS-CBH-EDR contains layered and total types of data and quality flags for cloud Base Height	474-00448-02-16_JPSS-DD-Vol-II-Part-16

Output	Description	Reference Document
VIIRS Cloud Layer Type	VIIRS-CCL-EDR contains layered and total types of data and quality flags for Cloud Cover layer	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Effective Particle Size	VIIRS-CEPS-EDR contains layered and total types of data and quality flags for Cloud Effective Particle Size	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Optical Thickness	VIIRS-COT-EDR contains layered and total types of data and quality flags for Cloud Optical Thickness	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Top Height	VIIRS-CTH-EDR contains layered and total types of data and quality flags for Cloud Top Height	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Top Pressure	VIIRS-CTP-EDR contains layered and total types of data and quality flags for Cloud Top Pressure	474-00448-02-16_JPSS-DD-Vol-II-Part-16
VIIRS Cloud Top Temperature	VIIRS-CTT-EDR contains layered and total types of data and quality flags for Cloud Top Temperature	474-00448-02-16_JPSS-DD-Vol-II-Part-16
Data Quality Notification for CBH	VIIRS-CBH-EDR-DQN indicate the test(s) that failed and the value of the DQN attribute of CBH	474-00448-02-01_JPSS-DD-Vol-II-Part-1
Data Quality Notification for CCL	VIIRS-CCL-EDR-DQN indicate the test(s) that failed and the value of the DQN attribute of CCL	474-00448-02-01_JPSS-DD-Vol-II-Part-1
Data Quality Notification for CEPS	VIIRS-CEPS-EDR-DQN indicate the test(s) that failed and the value of the DQN attribute of CEPS	474-00448-02-01_JPSS-DD-Vol-II-Part-1
Data Quality Notification for COT	VIIRS-COT-EDR-DQN indicate the test(s) that failed and the value of the DQN attribute of COT	474-00448-02-01_JPSS-DD-Vol-II-Part-1
Data Quality Notification for CTH	VIIRS-CTH-EDR-DQN indicate the test(s) that failed and the value of the DQN attribute of CTH	474-00448-02-01_JPSS-DD-Vol-II-Part-1
Data Quality Notification for CTP	VIIRS-CTP-EDR-DQN indicate the test(s) that failed and the value of the DQN attribute of CTP	474-00448-02-01_JPSS-DD-Vol-II-Part-1
Data Quality Notification for CTT	VIIRS-CTT-EDR-DQN indicate the test(s) that failed and the value of the DQN attribute of CTT	474-00448-02-01_JPSS-DD-Vol-II-Part-1

2.2.2 Algorithm Processing

The objective of the GCE module is to classify cloud pixels into layers and to generate cloud products (i.e., COT, EPS, CTT, CTH, CTP (Pressure), CBH, CC fraction, and CT) for these

layers averaged over horizontal cells. The module uses, as input, the pixel-level identification of cloud layer produced by the CCL module and the CCL aggregation table to create the products. See Section 2.5 of the Cloud Cover/Layers ATBD, D0001-M01-S01-014, for more information. The dataflow model for the GCE algorithm is illustrated in Figure 6.

Figure 6. Conceptual Process for Generate Cloud EDRs

GCE reads and processes data one scan at a time. Horizontal product cells are always comprised of pixels within a single scan. The assignment of pixels to product cells is accomplished using an aggregation table. For each scan line, GCE loops over all product cells and generates outputs based on the pixel-level cloud IPs identified with each cell. GCE uses a routine called `getScan()` to read the input data. The aggregation table is read using `loadAggTbl()` and the pixels associated with each cell are identified with `getAggCell()`. The GCE routine requires only the current scan.

2.2.2.1 Retrieval Logic

The main processing of the GCE module is carried out in the `aggregate()` routine. This routine takes pixel-level inputs for each product cell as identified by the aggregation table and computes average values for each cell and for each cloud layer identified in the cell. The routines, `average2d()` and `average3d()`, process each of the cloud products (COT, EPS, CTT, CTH, CTP (Pressure), and CBH) to produce the result for each cell and for each layer in each cell. These routines include logic to handle missing data. The averaging routines also ensure that only pixels belonging to the product cell, and identified with the current scan, are included. Separate routines, `QF_Lyr_2d()`, `QF_Lyr_3d()`, `QF_Lyr()`, are used to process IP quality data. At this point a correction from geopotential to geometric height is also applied to the CBH and CTH products. The CC and CT products are copied directly from the cell-averaged products supplied by the CCL algorithm.

2.2.2.2 Main Module – GCE_main()

`GCE_main()` is the main driver for the core GCE algorithm. Called by `ProEdrViirsGce()`, `GCE_main()` loads the aggregation table via `loadAggTbl()`, assigns scan pointers and processes granule data one scan at a time. For each cell in a scan, `getAggCell()` gets cell information from the aggregation table and `Aggregate()` aggregates pixel data into cells. After all scans have been processed, CCL IP Layer and Total fields are copied directly to the output EDRs, layered fields are flipped to obtain top-to-bottom stacking of clouds, and `ComputeGranuleQualityFlags()` is called to process granule quality flag data.

2.2.2.3 getScan()

getScan() assigns scan data pointers to the appropriate locations in the IP data using offsets calculated with pixel and cell position configuration information.

2.2.2.4 average2d()

average2d() is the averaging function for 2-dimensional data. It calculates the pixel average for the current cell and returns the result. Only valid pixels (i.e. assigned a cloud layer by CCL, determined to be product and current scan pixel according to the aggregation table, does not contain fill data) are used in calculations. If no valid pixels were processed, then a fill value is returned.

2.2.2.5 average3d()

average3d() is the averaging function for 3-dimensional (layered) data. It calculates the pixel averages for the current cell layers and returns the results. Only valid pixels (i.e. assigned a cloud layer by CCL, determined to be product and current scan pixel according to the aggregation table, does not contain fill data) are used in calculations. If no valid pixels were processed for a layer, then a fill value is returned for that layer.

2.2.2.6 Aggregate()

Aggregate() calls the averaging functions for data (average2d(), average3d()) and quality flags (QF_Lyr_3d(), QF_Lyr_2d(), QF_2d()) to process pixel data into cells and saves the outputs to the output structures. CBH and CTH data is converted to geometric height (Height_Conversion()) prior to being stored in output structures.

2.2.2.7 Height_Conversion()

Height_Conversion() converts geopotential to geometric height. This function requires latitude (in radians) and formula constants (c1, c2, c3) for processing.

2.2.2.8 posOutdata()

posOutdata () assigns scan data pointers to the appropriate locations in the output EDRs using offsets calculated with pixel and cell position configuration information.

2.2.2.9 put_4_level()

put_4_level(), called during quality flag processing, calculates the ratio data/count to determine which of the four categories the ratio falls within (0: 0 to <25%, 1: 25 to <50%, 2: 50 to <75%, 3: 75 to 100%) and sets the appropriate bit positions in the output flag using the offset supplied.

2.2.2.10 put_dominant()

put_dominant(), called during quality flag processing, calculates the ratio data/count to determine which of the two categories the ratio falls within (0: 0 - 50%, 1: 50 - 100%) and sets the appropriate bit positions in the output flag using the offset supplied.

2.2.2.11 QF_Lyr_2d()

QF_Lyr_2d() is the averaging function for 2-dimensional quality flag data and it gathers granule level quality data. After accumulating quality flag data for the current cell, put_4_level() and put_dominant() are called to average the data. The results are then copied to all the EDRs. For out-of-bound and algorithm-branching quality flags that are EDR specific, the previously stored flags are overwritten.

Only valid pixels (i.e. assigned a cloud layer by CCL, determined to be product and current scan pixel according to the aggregation table, does not contain fill data) are used in calculations.

If no valid pixels were processed, no quality flags are set for the current cell.

2.2.2.12 QF_Lyr_3d()

QF_Lyr_3d() is the averaging function for 3-dimensional (layered) quality flag. After accumulating quality flag data for the current cell, put_4_level() and put_dominant() are called to average the data per cell layer. The results are then copied to all the EDRs. For out-of-bound and algorithm-branching quality flags that are EDR specific, the previously stored flags are overwritten.

Only valid pixels (i.e. assigned a cloud layer by CCL, determined to be product and current scan pixel according to the aggregation table, does not contain fill data) are used in calculations. If no valid pixels were processed, no quality flags are set for the current cell layers.

2.2.2.13 QF_2d()

QF_2d() is the averaging function for 2-dimensional non-cloud-related quality flags. After accumulating quality flag data for the current cell, put_4_level() is called to average the data. The results are then copied to all the EDRs.

Only valid pixels (i.e. assigned a cloud layer by CCL, determined to be product and current scan pixel according to the aggregation table, does not contain fill data) are used in calculations. If no valid pixels were processed, no quality flags are set for the current cell.

2.2.2.14 sortLyrOrder()

sortLyrOrder() flips the layer order from bottom up to top down. Then, based on the CBH EDR product, it copies the rest of the EDR products in the similar sort.

2.2.2.15 sort_lyrs()

sort_lyrs() sorts an array with NLAYERS elements in descending order and indexes them. This routine is called within sortLyrOrder() routine.

2.2.2.16 loadAggTbl()

See Section 2.1.2.15.

2.2.2.17 getAggCell()

See Section 2.1.2.16.

2.2.3 Graceful Degradation

The GCE module does not implement graceful degradation. GCE retrievals rely entirely on the availability of data from the VIIRS cloud IPs. No logic is included to replace missing data with secondary sources.

2.2.4 Exception Handling

A mechanism for external termination of the algorithm has been implemented, called a stop callback. If a stop callback is issued, processing is terminated and no outputs are produced.

Error-handling in the Input (I) and Output (O) stages of the I-P-O algorithm addresses errors associated with reading/ writing of databases. If an error occurs with required inputs, the error is reported, the process is terminated and no outputs are produced. For optional inputs, a failure to get VIIRS Cloud Mask results in no cloud cover metadata being reported, and a failure to get a Data Quality Threshold Table results in that particular set of Data Quality Threshold Tests not being executed and no Data Quality Notification output for that set of tests is produced.

Error-handling in the Processing (P) stage involves granule level and cell level errors. All outputs are initialized with Not Applicable (NA) FILL (except quality flags which are initialized to zero) so all unprocessed cell level data contain FILL values.

For granule level errors, i.e. errors resulting from loading the aggregation table or an invalid number of scans, the error is reported, the process is terminated, and no outputs are produced.

For cell level errors, i.e. errors resulting from loading aggregation table cell data, the errors are reported and processing continues with the next cell.

2.2.5 Data Quality Monitoring

Each algorithm uses specific criteria contained in a Data Quality Threshold Table (DQTT) to determine when a Data Quality Notification (DQN) is produced. The DQTT contains the threshold used to trigger the DQN as well as the text contained in the DQN. If a threshold is met, the algorithm stores a DQN in DMS indicating the test(s) that failed and the value of the DQN attribute. For more algorithm specific detail refer to 474-00448-02-01_JPSS-DD-Vol-II-Part-1 and 474-00448-02-16_JPSS-DD-Vol-II-Part-16.

2.2.5.1 Quality Flags

The cloud processing chain has defined many QFs. The QFs of CCL at IP level are essentially the same as those defined in COP IP product. The GCE module computes the QFs of the EDR products based on the QFs of the IPs for the pixels making up each cell. The values of the QF may be represented either by a four-level classification, or a one-level indicated by either zero or one. In a four-level QF, the value of the QF is set to 0-3, corresponding to 0 to <25%, 25 to <50%, 50 to <75% and 75 to <100%. The one-level QF can have values either zero or one,

with zero being less than 50% quality. The four-level flag is calculated by the function `put_4_level()` and the one-level flag is calculated by the function `put_dominant()`.

Since each cloud EDR output, e.g., COT, is reported both in layered values and values averaged over the entire HC, three types of EDR QFs are stored in all cloud EDR outputs. They are 3-D “layer” flags, 2D “total” flags (by combining all layers), and non-layered “EDR” QFs for the HC, generated by functions `QF_Lyr_3d()`, `QF_Lyr_2d()`, and `QF_2d()`, respectively.

2.2.6 Computational Precision Requirements

The GCE module employs single precision throughout, with one exception. Scale information is stored as single precision in the binary files and must be read as such in the code.

2.2.7 Algorithm Support Considerations

See Section 2.1.7.

2.2.8 Assumptions and Limitations

The GCE retrieval algorithms assume that the VIIRS 750m SDR auxiliary data, the VCM EDR including Cloud Phase, the COP IP, CTP (Parameters) IP, and CBH IP are all available for processing. All inputs are expected in binary format at M-band pixel resolution.

The GCE QC flags (confidence flags) are derived from the values present in the QC for the individual cloud IPs.

3.0 GLOSSARY/ACRONYM LIST

3.1 Glossary

Table 8 contains terms most applicable for this OAD.

Table 8. Glossary

Term	Description
Algorithm	A formula or set of steps for solving a particular problem. Algorithms can be expressed in any language, from natural languages like English to mathematical expressions to programming languages like FORTRAN. On JPSS, an algorithm consists of: <ol style="list-style-type: none"> 1. A theoretical description (i.e., science/mathematical basis) 2. A computer implementation description (i.e., method of solution) 3. A computer implementation (i.e., code).
Algorithm Engineering Review Board (AERB)	Interdisciplinary board of scientific and engineering personnel responsible for the approval and disposition of algorithm acceptance, verification, development and testing transitions. Chaired by the Data Process Algorithm Lead, members include representatives from STAR, DPES, IDPS, and Raytheon.
Algorithm Verification	Science-grade software delivered by an algorithm provider is verified for compliance with data quality and timeliness requirements by Algorithm Team science personnel. This activity is nominally performed at the GRAVITE facility. Delivered code is executed on compatible GRAVITE computing platforms. Minor hosting modifications may be made to allow code execution. Optionally, verification may be performed at the Algorithm Provider's facility if warranted due to technical, schedule or cost considerations.
Ancillary Data	Any data which is not produced by the JPSS System, but which is acquired from external providers and used by the JPSS system in the production of JPSS data products.
Auxiliary Data	Auxiliary Data is defined as data, other than data included in the sensor application packets, which is produced internally by the JPSS system, and used to produce the JPSS deliverable data products.
EDR Algorithm	Scientific description and corresponding software and test data necessary to produce one or more environmental data records. The scientific computational basis for the production of each data record is described in an ATBD. At a minimum, implemented software is science-grade and includes test data demonstrating data quality compliance.
Environmental Data Record (EDR)	<p><i>[IORD Definition]</i> Data record produced when an algorithm is used to convert Raw Data Records (RDRs) to geophysical parameters (including ancillary parameters, e.g., cloud clear radiation, etc.).</p> <p><i>[Supplementary Definition]</i> An Environmental Data Record (EDR) represents the state of the environment, and the related information needed to access and understand the record. Specifically, it is a set of related data items that describe one or more related estimated environmental parameters over a limited time-space range. The parameters are located by time and Earth coordinates. EDRs may have been resampled if they are created from multiple data sources with different sampling patterns. An EDR is created from one or more JPSS SDRs or EDRs, plus ancillary environmental data provided by others. EDR metadata contains references to its processing history, spatial and temporal coverage, and quality.</p>
Operational Code	Verified science-grade software, delivered by an algorithm provider and verified by GRAVITE, is developed into operational-grade code by the IDPS IPT.
Operational-Grade Software	Code that produces data records compliant with the System Specification requirements for data quality and IDPS timeliness and operational infrastructure. The software is modular relative to the IDPS infrastructure and compliant with IDPS application programming interfaces (APIs) as specified for TDR/SDR or EDR code.

Term	Description
Raw Data Record (RDR)	<p><i>[IORD Definition]</i> Full resolution digital sensor data, time referenced and earth located, with absolute radiometric and geometric calibration coefficients appended, but not applied, to the data. Aggregates (sums or weighted averages) of detector samples are considered to be full resolution data if the aggregation is normally performed to meet resolution and other requirements. Sensor data shall be unprocessed with the following exceptions: time delay and integration (TDI), detector array non-uniformity correction (i.e., offset and responsivity equalization), and data compression are allowed. Lossy data compression is allowed only if the total measurement error is dominated by error sources other than the data compression algorithm. All calibration data will be retained and communicated to the ground without lossy compression.</p> <p><i>[Supplementary Definition]</i> A Raw Data Record (RDR) is a logical grouping of raw data output by a sensor, and related information needed to process the record into an SDR or TDR. Specifically, it is a set of unmodified raw data (mission and housekeeping) produced by a sensor suite, one sensor, or a reasonable subset of a sensor (e.g., channel or channel group), over a specified, limited time range. Along with the sensor data, the RDR includes auxiliary data from other portions of JPSS (space or ground) needed to recreate the sensor measurement, to correct the measurement for known distortions, and to locate the measurement in time and space, through subsequent processing. Metadata is associated with the sensor and auxiliary data to permit its effective use.</p>
Retrieval Algorithm	A science-based algorithm used to 'retrieve' a set of environmental/geophysical parameters (EDR) from calibrated and geolocated sensor data (SDR). Synonym for EDR processing.
Science Algorithm	The theoretical description and a corresponding software implementation needed to produce an NPP/JPSS data product (TDR, SDR or EDR). The former is described in an ATBD. The latter is typically developed for a research setting and characterized as "science-grade".
Science Algorithm Provider	Organization responsible for development and/or delivery of TDR/SDR or EDR algorithms associated with a given sensor.
Science-Grade Software	Code that produces data records in accordance with the science algorithm data quality requirements. This code, typically, has no software requirements for implementation language, targeted operating system, modularity, input and output data format or any other design discipline or assumed infrastructure.
SDR/TDR Algorithm	Scientific description and corresponding software and test data necessary to produce a Temperature Data Record and/or Sensor Data Record given a sensor's Raw Data Record. The scientific computational basis for the production of each data record is described in an Algorithm Theoretical Basis Document (ATBD). At a minimum, implemented software is science-grade and includes test data demonstrating data quality compliance.
Sensor Data Record (SDR)	<p><i>[IORD Definition]</i> Data record produced when an algorithm is used to convert Raw Data Records (RDRs) to calibrated brightness temperatures with associated ephemeris data. The existence of the SDRs provides reversible data tracking back from the EDRs to the Raw data.</p> <p><i>[Supplementary Definition]</i> A Sensor Data Record (SDR) is the recreated input to a sensor, and the related information needed to access and understand the record. Specifically, it is a set of incident flux estimates made by a sensor, over a limited time interval, with annotations that permit its effective use. The environmental flux estimates at the sensor aperture are corrected for sensor effects. The estimates are reported in physically meaningful units, usually in terms of an angular or spatial and temporal distribution at the sensor location, as a function of spectrum, polarization, or delay, and always at full resolution. When meaningful, the flux is also associated with the point on the Earth geoid from which it apparently originated. Also, when meaningful, the sensor flux is converted to an equivalent top-of-atmosphere (TOA) brightness. The associated metadata includes a record of the processing and sources from which the SDR was created, and other information needed to understand the data.</p>

Term	Description
Temperature Data Record (TDR)	<p><i>[IORD Definition]</i> Temperature Data Records (TDRs) are geolocated, antenna temperatures with all relevant calibration data counts and ephemeris data to revert from T-sub-a into counts.</p> <p><i>[Supplementary Definition]</i> A Temperature Data Record (TDR) is the brightness temperature value measured by a microwave sensor, and the related information needed to access and understand the record. Specifically, it is a set of the corrected radiometric measurements made by an imaging microwave sensor, over a limited time range, with annotation that permits its effective use. A TDR is a partially-processed variant of an SDR. Instead of reporting the estimated microwave flux from a specified direction, it reports the observed antenna brightness temperature in that direction.</p>
Model Validation	The process of determining the degree to which a model is an accurate representation of the real-world from the perspective of the intended uses of the model.
Model Verification	The process of determining that a model implementation accurately represents the developer's conceptual description and specifications.

3.2 Acronyms

Table 9 contains the acronyms most applicable for this OAD.

Table 9. Acronyms

Acronym	Description
ACO	Atmospheric Correction over Ocean
ADCS	Advanced Data Collection System
AFM	Airborne Fluxes and Meteorology Group
AOS	Acquisition of Signal
CBH	Cloud Base Height
CC	Cloud Cover
CCL	Cloud Cover/Layers
CDA	Command and Data Acquisition
CDR	Climate Data Records
CEPS	Cloud EPS
CI	Configured Item
COMSAT	Communications Satellite
COP	Cloud Optical Properties
COT	Cloud Optical Thickness
CT	Cloud Type
CTH	Cloud Top Height
CTP	Cloud Top Parameters
CTP	Cloud Top Pressure
CTT	Cloud Top Temperature
DES	Digital Encryption System
DHN	Data Handling Node
DQTT	Data Quality Test Table
EOS	End Of Scan
EPS	Effective Particle Size
ERBS	Earth Radiation Budget Suite
ESD	Electrostatic Discharge
EUMETSAT	European Organization for the Exploitation of Meteorological Satellites
FEDR	Full resolution Environmental Data Records
FMH	Federal Meteorological Handbook
GCE	Generate Cloud EDR (or Grid Cloud EDR)
GPS	Global Positioning System
GSE	Ground Support Equipment
HC	Horizontal Cells
HRD	High Rate Data
IGS	International GPS Service
IJPS	Initial Joint Polar System
IOC	Initial Operational Capability
IP	Intermediate Product
LEO&A	Launch, Early Orbit, & Anomaly Resolution
LOS	Loss of Signal
LRD	Low Rate Data
LST	Local Solar Time
LUT	Look-Up Table or Local User Terminal
Metop	Meteorological Operational Program

Acronym	Description
MSS	Mission System Simulator
NA	Non-Applicable
NCA	National Command Authority
NDT	Nitrate-Depletion Temperature
OAD	Operational Algorithm Description Document
OC/C	Ocean Color/Chlorophyll
PIP	Program Implementation Plan
PMT	Portable Mission Terminal
POD	Precise Orbit Determination
PPC	Perform Parallax Correction
RSR	Remote Sensing Reflectance
S&R	Search and Rescue
SCA	Satellite Control Authority
SDE	Selective Data Encryption
SDP	Software Development Plan
SDR	Sensor Data Records
SDS	Science Data Segment
SGI®	Silicon Graphics, Inc.
SI	International System of Units
SN	NASA Space Network
SOC	Satellite Operations Center
SPA	Sub-Pixel Aggregation
SRD	Sensor Requirements Documents
SS	Space Segment
SST	Sea Surface Temperature
TBD	To Be Determined
TBR	To Be Resolved
TBS	To Be Supplied
TEMPEST	Telecommunications Electronics Material Protected from Emanating Spurious Transmissions
TOA	Top of the Atmosphere
USB	Unified S-band
UTC	Universal Time Coordinated
VCM	VIIRS Cloud Mask

4.0 OPEN ISSUES

Table 10. TBXs

TBX ID	Title/Description	Resolution Date
None		