JHT First Year Report
August 1, 2003-May 1, 2004
Implementation of the Advanced Objective Dvorak Technique (AODT)
Tropical Cyclone Intensity Estimation Algorithm

PIs: James P. Kossin (CIMSS) and Christopher S. Velden (CIMSS)

Co-Investigator: Timothy L. Olander (CIMSS)

TPC Contacts: Jiann-Gwo Jiing, Jack Beven, Ed Rappaport, Rick Knabb, Jaime Rhome, Michelle Mainelli

Summary of Objectives: The primary goal of this project is the implementation of an objective algorithm for estimating tropical cyclone (TC) intensity from geostationary satellite data developed at UW-CIMSS, into the Tropical Prediction Center (TPC) operational data analysis platform N-AWIPS. The Advanced Objective Dvorak Technique (AODT) was developed primarily at UW-CIMSS on McIDAS, and is a state-of-the-art method designed to objectively estimate TC intensity from IR imagery. In recent years, the TPC has had access to AODT developmental code, and experimentally operated the AODT locally. This project is aimed at 1) implementing the AODT into the fully-operational TPC computing environment, 2) helping to evaluate and maintain the algorithm performance, and 3) providing subsequent updates to the N-AWIPS code when new and successful research modifications are demonstrated. The detailed plan for the integration of the AODT into N-AWIPS as agreed upon by the NOAA/Computing Development Branch (CDB), TPC and UW-CIMSS is attached as an Appendix.

First Year Achievements/Milestones: At this point, we have been successful at achieving all of our proposed milestones for the first year of this project. Here are the details of our accomplishments:

I) We have rewritten the AODT code so that it fits into the CDB N-AWIPS architecture as specified by CDB staff. This required us to recreate the AODT as a library function, and to create new library functions that allow the AODT to talk with the NAWIPS NMAP application without having to directly interact with this AODT library. This is what is known as the application program interface (API). CDB and CIMSS staff planned and subsequently accomplished three different steps in this process:

1) A preliminary delivery on 1 February 2004 which defined many of the API functions to pass variables between the NMAP application and the AODT library.
2) A second delivery on 1 March 2004 that focused on the interfacing of data transferred across the API (satellite data, intensity reports, error messages, and history file interaction).
3) A final delivery on 23 March 2004 (about two weeks ahead of schedule), which focused on the actual running of the AODT. For this release, we included all required code specified by the CDB programmers along with a Programmer's Manual to help them with

some of the coding issues, error codes, subroutine interfaces, flow charts, and other specific AODT information that they may need. This manual was requested by CDB.

There has been dialog between CDB and CIMSS staff since the final delivery in order to help with some loose ends and issues that emerged at CDB while compiling and integrating the code, but nothing that required a new release of the code. There was a relatively minor issue at CDB related to the reading and interfacing of the satellite data on the CDB system with the AODT library functions. This was discussed with David Plummer (CDB) and a solution was found.

Chris Velden, one of the UW-CIMSS PIs, visited TPC in April. The AODT implementation strategy was discussed with TPC personnel, and the first version of the N-AWIPS AODT Graphical User Interface (GUI) was briefly demonstrated. Suggestions were made to include a graphical representation of the AODT time/history file in subsequent versions of the GUI.

II) Integration of the AODT algorithm onto NMAP at CDB occurred on 15 April 2004, and testing has begun. In our most recent and advanced testing, image-by-image comparisons have been made between the NMAP AODT version at CDB and the McIDAS version at CIMSS, to make sure the results are the same. This testing is presently in progress.

The next expected milestone should occur around 15 May 2004 and signals the completion of NMAP-AODT testing and evaluation by CDB and TPC. The following milestone should occur on 15 June 2004 and signals the delivery of the NMAP-AODT to TPC within N-AWIPS v5.7.3.

Considerations for Year 2: Based on the successes of the CIMSS and CDB staff in achieving the Year-1 goals of this program, no deviation in the Year-2 timeline is expected and the Year-2 budget is unchanged from our original proposal.

The potential for completing the testing and evaluation process and providing the stated deliverables by the end of the second year is very high. Both the testing and evaluation will require substantial time and effort from CIMSS staff as well as CDB, and ultimately TPC staff, as reasonably reflected in the Year-2 budget.

=====================

Appendix

Plan for the Integration of the AODT Algorithm in N–AWIPS
December 19, 2003

This document summarizes the roles and responsibilities for the software integration of the Advanced Objective Dvorak Technique (AODT) into N–AWIPS. The players include the technique developers at the Cooperative Institute for Meteorological Satellite Studies (CIMSS), the N–AWIPS developers in the Computing Development Branch (CDB) at the National Centers for Environmental Prediction (NCEP), and the primary customer, the Tropical Prediction Center (TPC) within NCEP.

The overall objective is to facilitate future AODT enhancements into N–AWIPS for operational use at the TPC. This work is critical to ensure the success of AODT research and development into operations.

Focal Points and Contact Information

Steve Schotz	program focal pt (CDB)	steve.schotz@noaa.gov	301.763-8000x7186
David Plummer	technical focal pt (CDB)	david.plummer@noaa.gov	301.763-8000x7150
Michelle Mainelli	program & tech focal pt (TPC)	michelle.m.mainelli@noaa.gov	305.229-4421
Christopher Velden	program focal pt (CIMSS)	chris.velden@ssec.wisc.edu	608.265-8005
Timothy Olander	technical focal pt (CIMSS)	tim.olander@ssec.wisc.edu	608.265-8005

Roles

CIMSS and the CDB will jointly define an application program interface (API). This interface will allow any high-level application to execute the AODT algorithm as well as perform various AODT history file management. A technical meeting between the two groups took place on 15 and 16 December 2003 at NCEP where the initial API was developed. The API involves the creation of a set of “callable” interface routines and is described more fully in the section titled “Technical Approach”.

The role of CIMSS will be to re-organize the current AODT software (version 6.3) into the API form. CIMSS will undertake this effort immediately and provide periodic informal software deliveries to the CDB for testing and evaluation.

Concurrent with the CIMSS effort, the CDB will prepare the N–AWIPS NMAP application for the new AODT library integration. This work will involve (in conjunction with the TPC) the design and implementation of a graphical user interface that will allow TPC forecasters and specialists to perform the AODT analysis. The CDB will also consult with the CIMSS developer whenever necessary in order to expedite the AODT library development.

The role of the TPC will be to develop a plan for the testing of the NMAP implementation of the AODT to ensure that the software reorganization and integration has been performed correctly. This will involve the comparison of NMAP results with the current McIDAS AODT implementation.

The completed AODT library, including the source code, will become part of N–AWIPS distributions to the NCEP Service Centers and shall be freely distributed as part of N–AWIPS.

Proposed Tasks and Completion Dates

The work described above will be undertaken as follows:

- Present - AODT version 6.3 software reorganization begins; NMAP GUI (approved by TPC) development begins.
- 01 Feb 04 - Preliminary AODT library delivery to CDB containing all information exchange functions and history file functions within a unit testing module
- 15 Feb 04 - NMAP GUI skeleton complete with stub calls to the appropriate AODT library functions (delivered in N–AWIPS release v5.7.2).
- 15 Mar 04 - Intermediate AODT library delivery containing all remaining AODT functions
- 1 Apr 04 - Final AODT library delivery to CDB w/ all functions and complete unit testing module.
- 15 Apr 04 - NMAP integration complete; NMAP testing by CDB begins; sample NMAP executable sent to TPC for non-operational testing and evaluation.
- 15 May 04 - NMAP AODT testing and evaluation by CDB and TPC complete.
- 15 June 04 - AODT delivered in release v5.7.3.

Fallback Position

If, for whatever reason, the integration of the AODT v6.3 algorithm cannot be completed by N–AWIPS release v5.7.2, TPC operations will continue to run the N–AWIPS implementation of ODT version 5.3. This implementation is a stand-alone program commonly known as 'nodt' and is currently available to TPC forecasters via a script interface.

Technical Approach

The following sections (Interface, Error Handling and Information Hiding) define a preliminary specification of the services and functionality of the ODT library. All functions will be written in the C programming language, be thoroughly documented, and undergo unit testing (an example of which will be provided by CDB, see below).

- **Interface (part 1) - AODT library interface with a brief description:**

Public library functions - information exchange:
```
odt_sethistory ( filename, &iret );               - set history file name & load in mem
odt_gethistory ( filename, &iret );               - get history file name
odt_setdates ( d1, t1, d2, t2, &iret );           - set date information
odt_getdates ( d1, t1, d2, t2, &iret );           - get date information
odt_setoptions ( dom, ic, land, srch, &iret );    - set options
odt_getoptions ( dom, ic, land, srch, &iret );    - get options
odt_setscene ( scenetype, &iret );                - set scene type
odt_getscene ( scenetype, &iret );                - get scene type
odt_setlocation ( lat, lon, &iret );              - set location lat,lon
odt_getlocation ( lat, lon, &iret );              - get location lat,lon
odt_setpaths ( topo, hist, auto, outp, &iret );   - set directory paths
odt_getpaths ( topo, hist, auto, outp, &iret );   - get directory paths
odt_setsst ( filename, &iret );                   - set SST BUFR file name
odt_getsst ( filename, &iret );                   - get SST BUFR file name
odt_getversion ( &version, &iret );                             - get ODT version as string
```
Public library functions - actions:
```
odt_loadIR ( temps, lats, lons, idim, jdim, &iret );  - load IR image information
```

```
odt_getnexthist ( &hist_struct, &iret );              - retrieve next history record
odt_listfmt ( hist_struct, &string, &iret );          - format hist struct as listing
odt_bullfmt ( hist_struct, &string, &iret );          - format hist struct as bulletin
odt_delete ( &iret );         - delete records within specified history file bounded by dates
odt_writehist ( &iret );                             - save latest computed results to history
                                      file
odt_autolocation ( &lat, &lon, &iret );               - compute storm center automatically
odt_getresults ( &string, &iret );                    - get latest results in string format
odt_part1 ( &iret );                                  - execute ODT algorithm, part 1
odt_part2 ( &iret );                                  - execute ODT algorithm, part 2
odt_getmessages ( &string, &iret );            - retrieve diagnostic messages from latest ODT run
odt_qerror ( error_num, &string, &iret );             - convert numeric error code to string
```

Exchange structure:
A C language structure will be defined by CIMSS to be used by an APPL to facilitate the exchange of extensive AODT history data. This structure will be defined by the LIB and located in the include file "odt_exch.h". This include file will contain only this structure definition. All other AODT library structures, definitions, etc., must be located in an include file private to the AODT library (see the section "Information Hiding" below).

Test Program:
A library test program shall be written to enable independent unit testing of the above functions. Elimination of the complexities of a high-level application allows for more efficient development and diagnostic testing. A sample of such a library test program is provided by CDB (attached).

· **Error Handling**

· All errors and potential errors must be trapped.
· Trapped errors should set an internal error message.
· Trapped errors must return control to the calling function, not exit.
· The calling application has the responsibility to check the return code(s) and take appropriate action. The library should not make assumptions about what action to take should an error occur.
· Completion codes:
Normal completion with no errors or warnings should return zero.
Completion with warnings and valid returned data should return a positive number.
Completion without valid returned data should return a negative number.

· **Information Hiding**

· If memory is allocated (malloc) it should be released within the same function call.
· No information in the library or include file(s) may be shared with or known by the application code. Examples: array sizes, parameter values, structures, etc.
· The library should not reference environmental variables; for instance, don't use `getenv' for directory paths, etc. For instance, for directory paths pass in this information via `odt_setpaths'.

- .#define - To avoid any confusion, local definitions and/or modifications to popular functions or parameters should be assigned unique names such as AODT_PI or AODT_SIN instead of PI and SIN.
- Text information normally sent to standard output should either be internally accumulated in a virtual file or character string, or directed to a physical file specified by the application (via odt_sopts). Several options exist for making the information available to the calling function.