An Algorithm Development and Support Database for MSTAR

Marc Boillot, Jose C. Principe
Computational NeuroEngineering Laboratory, University of Florida
Gainesville, FL 32611
{boillot,principe}@cnel.ufl.edu

Abstract

We have created an image processing GUI environment for algorithm development and database support. A fundamental requirement of efficiently and methodically documenting algorithm performance is keeping account of how results are created and having the ability to recreate those results at any time. The primary purpose of the ADSD (Algorithm Development and Support Database) software environment is to serve as a library development center for Moving and Stationary Target Acquisition and Recognition (MSTAR) algorithms and their associated results. This environment runs on NT workstations under the PV-Wave 6.10 platform. The objective of this database utility is to document the performance of registered algorithms for permanent storage with respect to: the user-adjustable algorithm parameters, the user-selected images processed, the date, and the results.

Review

Many software development teams use version control software to keep track of their code history and code revisions. Revision control is essential for development groups since it allows all the members to visualize the progress of the code and it allows them to work collectively without creating conflict in the code development stages. However, revision control is too restrictive for a research environment, where the members of the team work independently for long periods of time, and develop code with the goal of testing ideas which are not necessarily to be part of the final product. Therefore, we concluded that a more versatile environment was needed, preserving some of the important features of version control environments but molded to the idiosyncrasies of algorithm development.

The ADSD provides a work environment tailored to documenting algorithm performance. Algorithm performance is a measure of the algorithm's ability to generate a certain acceptable level of results for certain test scenarios. The ADSD maintains record of all algorithm versions used in testing/Training and the associated test/train images, parameters, date, and results by means of a GUI. It allows developers to register a functioning algorithm and declare user-adjustable algorithm parameters which are globally available to all other users that test or use the algorithm. Algorithm performance is dependent on how well the algorithm parameters are selected with respect to the application. Many applications for similar algorithms require fine tuning of the algorithm parameters to achieve comparable results. The difficulty in documenting algorithm performance is determining and remembering which tuning of the parameters gave the best results. A valuable investment in algorithm development is to document how the algorithm performs with respect to parameter variation as the algorithm is refined for a particular application. All algorithm code in the ADSD is documented with respect to its version. As changes are made to the algorithm code newer versions are launched and this allows the developers to see the progress of their code.

The ADSD is an interactive database which integrates algorithm development, result visualization, results storage and retrieval, and parameter association in a user friendly interface environment. The ADSD was initially based off the UWB-ATR evaluation environment created by the Electronic Com-
communications Laboratory (ECL) [3]. The ADSD resembles the UWB-ATR user interface and utilizes some of the display and database association functions of the UWB-ATR environment. The ADSD follows the structural organization of the GUI but it has a function and application quite different from that of the UWB-ATR environment. The UWB-ATR environment's achievement is in efficiently documenting algorithm detection results to terrain scenes. It displays target recognition performance by way of overlaying detection results on tested terrain scenes, through ROCs, and other visually informative displays and tables. Algorithm parameters can be adjusted to enhance detection performance but the output is restricted to follow a coordinate plot. No information in the algorithm processing stage is saved for reference or future use. The ADSD was created to document algorithm performance in the development phases since results are more than hit or miss. Algorithms usually extract feature information from the data sets before classification. This information decides the potential of the algorithm. The ADSD was developed to store any type of algorithm results using MSTAR images in the processing stage. It's function is to allow the developer to tune an algorithm for optimal results and to keep any relevant results created in the processing stage with the algorithm and the selected parameters.

**ADSD Environment**

The ADSD is loaded as a file directory onto a PC or NT workstation, and is evoke by calling an executable within PV-Wave. The ADSD program GUI uses all the family specific applications of PV-Wave and allows the developer to do so as well. The ADSD effectively acts as a shell between the MSTAR data and the developer's algorithms. The MSTAR data sets are linked to the ADSD by specifying the data set directory in the GUI. The large MSTAR datasets can then be kept on disc space independent of the operation of the ADSD. The ADSD provides a means of selectively choosing any MSTAR images to import to any registered algorithm by mouse point and click. Current algorithm development requires loading an image into a common variable or matrix and having the algorithm open the image for processing. The ADSD provides a link to the MSTAR image sets and does not require any responsibility from the developer to load the image. And, if desired, it automatically links the image files used to the selected algorithm for archiving. The ADSD essentially provides a window into all the images stored in the MSTAR database by showing the user all the available images as well as the header information associated with each image.

The same simplicity of mouse point and click to import images to an algorithm for testing is extended to reviewing algorithm results. All functioning algorithms are stored in a GUI file management window, in which all developers have been given a user directory to store their algorithms. Each developer works independently in the development stage until they feel their algorithm is acceptable for registration. The user then registers the algorithm to the ADSD within their user directory. Then it is available to the group to test or revise. Under each user directory each algorithm has its own directory which contains all the versions of the working algorithm as well as any results. All saved results are associated to the particular algorithm version that created those results along with the parameter values, the images used, and the date. Each version of an algorithm is linked to its own set of results. The ADSD associates all results to the particular algorithm that created them. Thus, the developer's responsibility of associating results to algorithms is no longer required. This is very valuable for any developer who understands the burden of documenting all results to versions of the same algorithm while keeping account of what results belong to which version. And, since the responsibility of the developer tends to increase as other algorithms are introduced, the ADSD relinquishes this check and balance requirement. The ADSD assumes this association responsibility completely and allows the developer to concentrate on algorithm performance and not documentation.

The ADSD GUI allows any user, developer, or supervisor to visualize all algorithm work, current or past, by means of a windows file management utility. This is a valuable tool for supervisors or project leaders to see the performance of registered algorithms and the directions that need to be taken to complete tasks. The fundamental contribution of the ADSD is its algorithm-results association functionality. The ADSD associates an algorithms performance to its results by recording all the details needed in generating those results. These details are: the user-adjusted parameters selected to fine
Registered algorithms in the database with their own, or other users' algorithms. The database automatically links the algorithm used, the adjustable parameters, and the images used on any particular run to the corresponding results file. To use the ADSD the developer only needs to follow a simple standard format specified for each of their algorithms routines.

**Required Algorithm Format**

The ADSD is a graphical user interface (GUI) that runs on a PV-Wave 6.10 platform. PV-Wave is a high performance language for image processing. The user interface was developed using the PV-Wave widget utilities toolkit and occupies under 20 Megs uncompressed. All the functions available in PV-Wave are passed through the environment and are available to the user in their algorithms. The algorithm format for the ADSD is different from the UWB-ATR environment. The ADSD requires the developer to partition his code into three main sections that will be used by the ADSD as independent processing functions. The ADSD is dynamic in the sense that it runs the algorithm sections from the GUI. Like the UWB-ATR environment, the ADSD can important results for display, but the difference is that it can run the stored algorithms itself. It does not require the developer to manually run the algorithm outside of the environment and generate results to submit or import. The ADSD allows the developer and other users to run their algorithms interactively within the environment. The ADSD has been created for the purpose of algorithm management and utility and to provide a work friendly environment for algorithm development. The GUI is a window shell into the available MSTAR images and registered algorithms. This database is an interactive algorithm support utility that allows developers to verify and/or compare

The main objective of this database is to associate user selected images to the user's algorithm. Processing an algorithm requires three main parts, 1) a set of images to process 2) a list of adjustable parameters to configure the main part of the algorithm, and 3) a means of displaying the processed results. An algorithm format has been proposed which allows the ADSD to interact with a developer's algorithm. The format specified in this section gives the database the capability of 'calling' any registered algorithm and applying it to any selected set of MSTAR image chips. This type of functionality is important if multiple sets of registered algorithms need to be compared on a common set of images. Algorithms are partitioned into three main parts for the ADSD format:

1) A set of user adjustable parameters predefined by the algorithm developer. Examples could be the number of hidden neurons in a neural net, the number of layers, the learning rate, the final error, and so on. The algorithm developer must decide what parameters he wants to declare as user adjustable.

2) A core processing part of the algorithm to read in a set of images and output some type of result or set of results. Inside the processing function the developer has the benefit of providing multiple window status display and screen prompting while the algorithm is running. For example when we
train a neural net we display a learning curve every so many epochs or display the adaptation error at the prompt to inform us of the training status.

3) A results display to show how the results are presented and what the output variables are that represent the results. The developer knows how the results should be displayed since it is their algorithm and what are the output variables. This is how the environment is able to display saved results from previous algorithm runs. The ADSD simply calls the results display routine to display any saved results generated by the processing procedure.

The algorithm must follow the mentioned guidelines to be usable in the database environment. The format is simply a partitioning of the developers algorithm code into the three listed parts mentioned above. If an algorithm has been written and performs properly the developer should be able to pull the user adjustable parameters to the top of the code and pull down the display results utilities to the bottom of the code. This is essentially the required format for this database. Once the code has been partitioned, the three sections need to be isolated as stand alone processes by declaring them as independent procedures. This is a trivial procedure in PV-wave. The format for the code is simply sectioning out these parts and declaring them as procedures. This allows the ADSD to call any of the three sections at a given time.

Once the algorithm is in the correct format and registered with the ADSD it is available to all users for testing or documentation. A user simply mouse points and clicks on an algorithm in a GUI file listing he wants to test. He is then prompted to mouse point and click on a set of MSTAR chips that will be sent to the selected algorithm. A window is then displayed which contains a list of the user adjustable algorithm parameters the developer declared in the registration of his algorithm. He adjusts the parameters according to his application or anticipated needs, and then clicks a RUN button. The ADSD then calls the selected algorithm with the new modified parameters for testing on the selected MSTAR image sets. Once the algorithm finishes processing, results are displayed according to how the algorithm developer wanted the results to be presented. The results can then be saved by simply clicking a save button and the ADSD automatically associates all the users selections to that algorithm. The results are then on file and are available for review through the GUI. In the event that a supervisor wants to see the results of a certain algorithm he simply clicks on a button to display the results. If he wants to re-run the algorithm he simply presses a button that calls the processing procedure of the algorithm and adjusts the parameters accordingly. If he wants to change the image sets he simply selects by mouse point and click which images he wants. Once the algorithms are registered with the ADSD, results can be easily generated on unseen images and newer algorithms versions can be launched. This allows supervisors to see the performance of registered algorithms and to guide its development and direction. The ADSD ensures that results are actual results, and that all results can be recreated exactly from the algorithm and the selected parameter values that generated those results.

Applications in Pose Estimation

The functionality of the database can be of use in the pose estimation problem, which has been investigated at our laboratory. It is of desirable interest in ATR research to estimate a given target’s aspect angle. The pose angle has been estimated using a novel mutual information algorithm [1], [2]. The pose can also be further utilized for vehicle classification. It was our research goal to evaluate the accuracy of the new pose estimator. Several variables where of interest. Since the algorithms learn the best projection from a training set, many possible variations of the training set were possible, such as: one or several vehicles, one single depression angle, or many different depression angles. Moreover, we would like to estimate the generalization of the algorithm in several mixture of test set
vehicles to assess performance. For each training and testing configuration, different algorithm parameters and algorithm versions needed to be tested. The number of possibilities runs very quickly on hundreds of tests. We had two different researchers conducting all these tests. The algorithms for pose estimation have been registered in the ADSD with little effort and testing runs have been completed on untrained data sets. With the ADSD we can quickly relate to performance the effect of parameters, training mix and algorithm version. More importantly, even once the researchers (which are normally Ph.D. students) leave the lab the results can be duplicated.

Conclusions

The ADSD is fully operational and is currently being employed in the CNEL lab. The ADSD provides a suitable environment for algorithm development and documentation. Refinements to the pose estimation algorithm using the ADSD secures result documentation due to the ADSD’s developer independent association functionality. In the future, the ADSD platform can be extended for use as a full target recognition development center. Currently, it’s primary purpose is to document algorithm performance for the specific recognition scheme on selected MSTAR target chips. The target chips can be used to essentially tune the registered algorithms for feature extraction on a particular chip set. The ADSD serves currently as a development database for MSTAR chips only and not yet for full terrain scenes. The long term objective is to have a library of algorithms which can be combined specifically for the cause of target classification and recognition. A full development platform would incorporate full target and clutter terrain scenes. The ADSD can be extended to this application which would allow the registered algorithms to be tested on terrain scenes. The effort to develop full target acquisition and recognition systems relies on the ability to record and document the individual stages of algorithm development as effectively and correctly as possible. The ADSD GUI platform allows for the complete development of documenting an algorithm’s performance and utility.

Acknowledgments: This work was partially supported by a DARPA grant F33615-97-1-1019.

References

