Using an image retrieval system for image data management

T.P. Karnowski[a], K.W. Tobin[a], R.K. Ferrell[a], W.B. Jatko[a], F. Lakhani[b]
[a]Oak Ridge National Laboratory, Oak Ridge TN; [b]International Sematech, Austin TX

ABSTRACT

Images of semiconductor defects are maintained in semiconductor yield-management systems to help diagnose problems that arise during the manufacturing process. A common problem in future systems is the number of images to maintain, which is increasing at an alarming rate due to the growing use of in-line and off-line imaging systems. A manufacturing-specific content-based image retrieval system, or Automated Image Retrieval (AIR) system, was developed by ORNL in coordination with International SEMATECH during 1998-1999. The system uses commercial databases to store image information and uses a customized indexing technology to rapidly retrieve similar images based on visual content. In addition to acting as a yield management tool based on storing and retrieving images, the system can be utilized as a tool for data management by helping determine when images are redundant in relation to previously stored data. Ideally this information can be used to time-stamp the data for future purging based on a variety of ratings such as "long-term", "mid-term", and "short-term". In some situations the feedback from the AIR system can even be used to omit the image entirely based on pre-existing close matches. In this paper we explore techniques for using the AIR system to assist in image data management. Experimental results are shown with simulated image data representing various degrees of image clustering or redundancy, and manufacturing image data accumulated during earlier field-testing of the AIR system at industry sites. Early results indicate substantial reductions in the size of industry databases may be achievable while continuing to maintain an adequate representation and history of the manufacturing process. To reduce the number of stored images, AIR technology can be used in place of, or as a guide for, the typical “store for N months and purge” approach to image management. This approach will enhance the use of the image database, since the real bottleneck in such a procedure is the need to sort such massive amounts of stored data as opposed to actual disk space.

Keywords: Content-based image retrieval, data management, defect detection

1. INTRODUCTION

Images of semiconductor defects are maintained in semiconductor yield-management systems to help diagnose problems that arise during the manufacturing process. A common problem in future systems is the number of images to maintain, which is increasing at an alarming rate due to the growing use of in-line and off-line imaging systems. Indeed, image management in semiconductor yield management systems is a growing cause of concern since many facilities collect 3,000 to 5,000 images each week, with future estimates of 12,000 to 20,000 per week commonly given. The vast majority of this data comes from instruments such as inspection tools (both in-line and off-line) including optical microscopes, laser-scattering systems, and SEMs [1].

A manufacturing-specific content-based image retrieval system, or Automated Image Retrieval (AIR) system, was developed by Oak Ridge National Laboratory (ORNL) in coordination with International SEMATECH during 1998-1999 [2]. The system uses commercial databases to store image information and uses a custom indexing technology to rapidly retrieve similar images based on visual content. The system has been utilized in field-testing at various International SEMATECH member companies during 2000, with a goal of demonstrating the premise that defect images with similar appearance have similar manufacturing sources [3].

The AIR technology has been demonstrated through this field-testing to provide the end-user with an efficient means for organizing pre-existing repositories of images and extracting useful information about historical process conditions that can be used to address current process issues. Another aspect of the AIR technology that has been developed relates to data management and intelligent data storage: specifically, minimize the number of data samples that must reside in the Defect Management System (DMS) while adequately representing the manufacturing process for future problem solving. Due to the rapid growth in the size of the fab image database and the high potential for redundancy, a method has been designed to reduce this redundancy and enhance either, (1) the long-term storage of the most information-rich image content (i.e., maintaining the same capacity but keeping selected data for a longer period of time), or (2) a

reduction in the size of the repository capacity which results in improved performance (i.e., storage and retrieval efficiency) and reduced time for re-indexing. Based on these two criteria, an image data management method should address the mitigation or elimination of redundant data in the DB, it should limit the number of entries to be maintained by the AIR indexing structure, and tag each entry with a lifetime for maintenance in the system based on a measure of redundancy (for example, delete after 1 month, 6 months, never delete).

In this paper we explore techniques for using AIR-based intelligent data management (IDM) to eliminate redundant images from an archive. We discuss our experimental configuration and the issues associated with performing such experiments. We then show experimental results with two sets of data: first, with simulated image data representing various degrees of known image clustering or redundancy, and second, with manufacturing image data accumulated during earlier field-testing of the AIR system at industry sites. We conclude the paper with some observations and summarize our results.

2. IMAGE RETRIEVAL SYSTEM

Figure 1 shows a schematic representation of the image indexing and query process [2]. Image indexing refers to the preliminary organization of the image data that must be accomplished prior to image retrieval through the query process. Our AIR software tool is an independent "engine" that works in conjunction with directories of image files. The indexing process begins by scanning all the available imagery in the database and building a table of image feature sets (also known as feature vectors), where each row in the table corresponds to one image. Several feature vectors are maintained for each image that describes independent characteristics of the scene. Once the table is generated, an indexing process takes place that organizes the data into a tree structure that allows for rapid retrieval of the imagery during a query. Both the feature vector table and the indexing structure are maintained for use during the user query process. As new images are added to the database, the feature vector table and the indexing structure are updated according to some schedule.

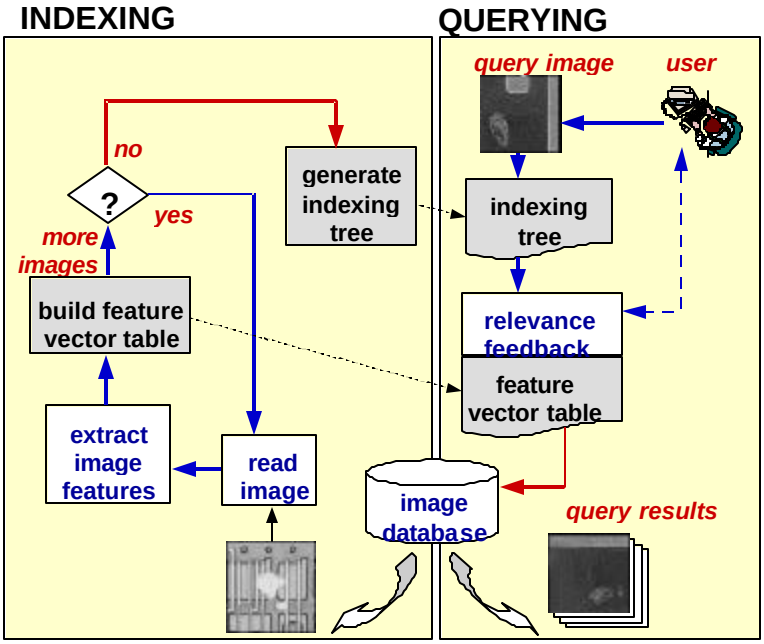

Figure 1 Schematic representation of the indexing and query process for Automated Image Retrieval

Once the feature vector table and indexing tree are established, the user (an individual or a computer process) performs a query by presenting the system with a new image. The image is submitted as a query after conversion to a series of feature vectors. By passing the query image through the tree, a subset of the images that shares similar feature characteristics is rapidly obtained. These results may be used by the user to refine the search as represented by the relevance feedback block in the diagram. The final result is a display of images that are similar to the user's query. These are displayed along with a similarity metric that indicates how close the results are to the original query image. Also associated with each retrieved image is a list of linked data

that ties the imagery to the fab. This data includes dates, lot numbers, wafer identification numbers, process layer, tooling, etc.

All images submitted to AIR are separated into a defect and background. This separation is performed through the use of existing defect masks generated by inspection tools or an automated defect segmentation process. In turn, four descriptive areas characterize the defect: shape, size, color, and texture. Two descriptive areas characterize the background: color and structure. These areas are used to categorize descriptive features that characterize the image visually and what is most important about the image to the users. Furthermore, this allows the retrieval process to find distinct combinations of characteristics; for example, it is possible to find certain defect types on different backgrounds, or to focus the queries on particular aspects of the defect and background (for example, queries can be conducted that ignore the defect color, the background color, the defect size and texture, etc.)

Another aspect of AIR that is important for the image management problem is the indexing tree structure. Figure 2 illustrates the indexing tree concept. The tree consists of decision points where different feature values are evaluated and decisions are made based on the values of the query vector. The net result of this structure is shown in the second part of the figure, where we see the partitioning of feature space by the decision tree into bins. We use the Approximate Nearest Neighbor algorithm [4] to assemble and query the binary decision tree. With the use of the ANN algorithm, an approximation factor e can be utilized to trade-off query precision with speed. At the crudest, the ANN algorithm simply returns every image in the same bin as the query image; at more refined levels, the algorithm examines neighboring bins to find images that are as likely a close match as those in the query image bin. There is a separate indexing tree for each area of interest. In the example below a two-dimensional feature space is illustrated.

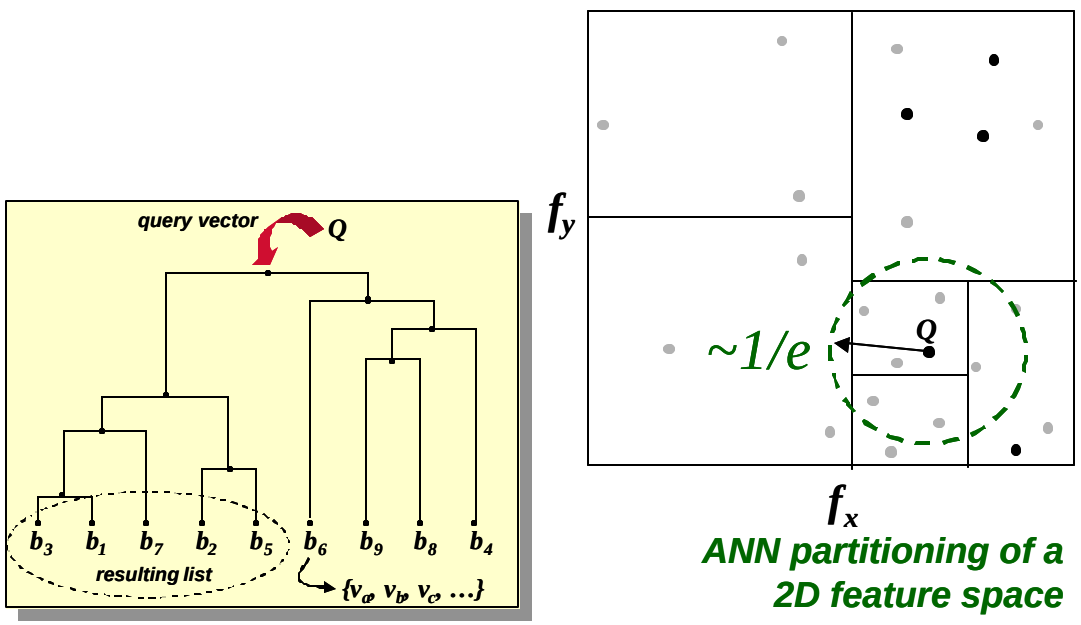

Figure 2 In AIR the images are organized by binary decision trees with an approximate nearest neighbor algorithm used in searching.

3. AIR AND IMAGE MANAGEMENT

As mentioned, the motivation behind the development of the AIR system was the need to effectively use the large and growing amounts of imagery obtained in semiconductor fabrication facilities. While the effectiveness of AIR in utilizing the large amount of data has been demonstrated, we began investigating the capability of AIR to "prevent the problem" in the first place: in addition to acting as a yield management tool based on storing and retrieving images, the AIR system can be utilized as a tool for data management by helping determine when images are redundant in relation to previously stored data. Ideally this information can be used to time-stamp the data for future purging based on a variety of ratings such as "long-term", "mid-term", and "short-term". In some situations the feedback from the AIR system can even be used to omit the image entirely based on pre-existing close matches.

We should point out that our objective is not to conserve memory on hard drives. Rather, we seek to show that, despite the growing amount of available disk space, AIR technology can be used in place of, or as a guide for, the typical "store for N months and purge" approach to image management. Reducing the amount of data saved has benefits in many ways, in that the existing data is more valuable because it has met some criteria for keeping and also is easier to use. Using the AIR approach will enhance the use of the image database, since the real bottleneck in such a procedure is the need to sort such massive amounts of stored data as opposed to actual disk space

Figure 3 shows a block diagram of an AIR IDM system. A candidate defect image, along with associated non-defect information (lot, layer, classification, etc) is submitted to the system as a query. We show the query with two components: an image-based query (using AIR) and a non-image query (using a "Defect Redundancy Engine", which could be as simple as a database query against the fab data management system). The results of the image-based query are fed to a "Redundancy Function", which can use the following methods (or some other technique) that seek to measure the similarity of the image to others in the database, resulting in a redundancy measure.

Use the statistics of the contents of the indexing tree bins

The bins shown in figure 2 are coarse approximations of the clusters of images in the database. A candidate image can be evaluated by locating its bin then determining how similar it is to the other images. This could be based on statistical measures like standard deviation or Euclidean distance measurements. We can take this strategy one step further by using this evaluation with all the indexing trees in the system.

Use a criterion based on the closest returned images

Queries can be conducted and the returned images evaluated in terms of confidence measures based on the Euclidean distances returned from the query. Images matched by several database images with a high confidence measure can be rejected as redundant.

From this point, the final functional block – a Keep / Reject decision function – is employed to decide to store the image long-term or reject it as redundant. The decision function may include other data about the returned images from the image-based query such as their lot numbers or layers.

We should note that we do not show that there should be some strategy to keep ALL images for some length of time. In addition, the system should also not be deployed at all until the image database reaches some level of maturity. Ultimately, the method chosen must try to eliminate sample imagery that is redundant or contains little valuable information for future diagnosis of problems. In this regard the last technique, combining a purely image-based method with auxiliary wafer information, adds considerable value.

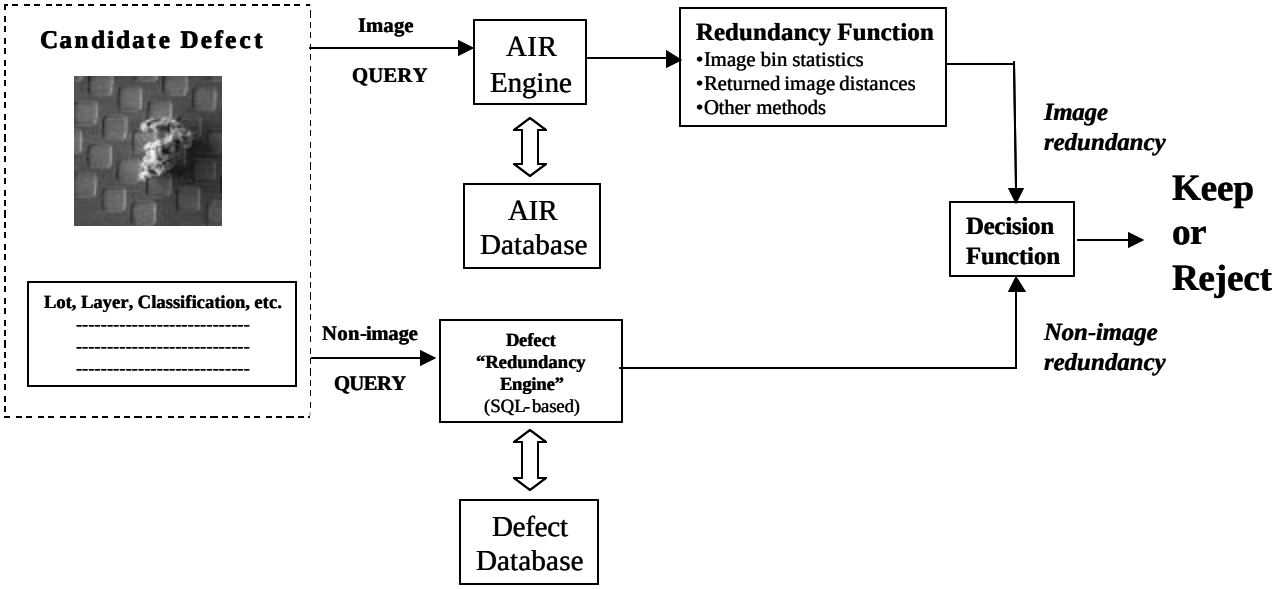

Figure 3 AIR intelligent data management system block diagram

4. EXPERIMENTAL CONFIGURATION

There are considerable difficulties in determining techniques for evaluating the performance of AIR in eliminating redundant imagery. The most daunting is trying to determine how an image is redundant. Ideally, an image would be deemed as redundant if other images in the database will allow the yield engineer to determine the root source of problems in the fab. An ideal trial of the AIR IDM system would be to employ two equally competent yield engineers over a period of one year, allowing one to use the complete image database and the other an archive screened by an AIR IDM system.

Unfortunately, such an ideal test is not possible for economic reasons. Instead, we sought to show that reducing the size of the database allowed us to achieve similar results to a full database. This is a simple test that verifies the fundamental premise of the work: AIR can be used in an IDM system to identify and eliminate redundancy in images that a DMS seeks to archive. In our experiments, we did not seek to actually implement the IDM system shown in figure 3. Our criterion for determining redundancy was analogous to using the returned image distances.

In our testing we used two sets of "images". The first set was a simulated image set generated by computing random values for image features. As explained in section 2, the AIR system works on image features – descriptive numbers that seek to measure characteristics that people utilize when making visual comparisons. We generated the random images by assigning a number of clusters with different mean values in multi-dimensional feature space, then assigning a standard deviation for the cluster and computing random "images". These random sets allowed us to study the problem with a known degree of redundancy, in that two images from the same cluster are regarded as redundant.

The second set consisted of data compiled during field-testing at a SEMATECH member company. Our tests were done in an analogous fashion, but in this situation the number of clusters, or classes, was defined by the category of the data. Each image had three different cluster IDs: the lot identification numbers (1,350 clusters), the layer identification numbers (99 clusters), and the optical classification codes (144 clusters).

Our testing procedure is as follows. A database of 50,000 images (synthetic or actual) was obtained. We removed a randomly selected sample set of 10,000 images to use solely as query images. We "seeded" the database with a set of 1,024 images. We then evaluated each remaining image as redundant with the 1,024 or not redundant. This redundancy

measure contained parameters to relax or tighten the "standard" for redundancy, effectively implementing the redundancy function and decision function blocks of Figure 3. When all 40,000 images had been submitted and added or rejected, we took the 10,000 held-out images and queried them against the database. The results of each query were saved and compared to the classification of the held-out image (cluster number for simulated data, and lot / layer / classification code for the field-test data) to obtain performance data. We then relaxed the redundancy parameters so that more similar images were regarded as "less redundant" and performed the test again. This procedure was repeated until all 40,000 images were added to the database. At that point we have a set of data showing performance at different threshold levels, which we plotted as database size vs. performance.

5. EXPERIMENTAL RESULTS

Regarding the simulated data, a set of random feature vectors were generated as Gaussian clusters in feature space. The selected parameters included: (1) the number of samples (i.e., feature vectors) in the dataset; (2) the number of samples held out for k-NN testing; (3) the number of defined features per vector; (4) the number of defined classes (i.e., clusters); (5) the class parameters including the class mean and standard deviation for each feature, and; (6) the size of the initial dataset to seed for the experiment.

An example of the results obtained for a dataset similar to the AIR field test data is shown in Figure 4. This figure shows three curves that were generated for the test cases using 6, 24, and 36 length feature vectors respectively. The size of the dataset was 50,000 data points (i.e., feature vectors), with 100 clusters and 10,000 held out (for a maximum database size of 40,000). The x-axis in this figure represents the fraction of the database that passed the threshold test described above. The y-axis is the performance measured after each pass through the data set with a given threshold level. For this case, the performance measurement was "1-Nearest Neighbor" or 1-NN, which means we took the closest returned image and used its classification as the "estimated" class. Note for the most complex case, i.e., 100 clusters and 36 features per point, the system only required 37% of the original data points to achieve a maximum classification performance. This verifies our assumptions regarding the mitigation of redundancy in the DB for a parametrically

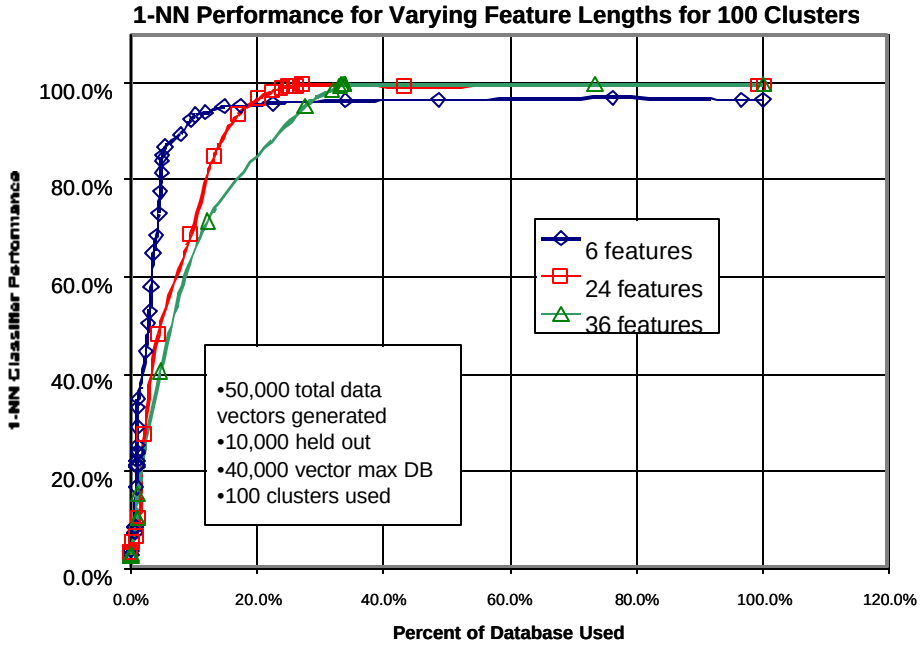

Figure 4 – Example showing data management results for a simulated dataset of 50,000 points. Note for the most complex case with 36 features per point that only 37% of the DB is required to be maintained.

controlled simulation, i.e., the data generated and used in Fig. 4.

The experimental image set used a data set of 50,000 images chosen randomly from the field-test set of 62,594. The field-test results showed that a 4-Nearest Neighbor (4-NN) comparison achieved an average performance of approximately 60% when using the lot ID, layer ID, and optical classification as the cluster identifier. (For the 4-NN measurement we took the four closest returned images and used their classifications as votes; the classification with the most votes was deemed the estimated cluster class.) For each result here, we normalized the 4-NN performance obtained with our growing database to show performance as a percent of "ideal" performance (i.e., the best performance possible using AIR as a cluster classifier.)

A plot of these results showing normalized 4-NN performance (chosen for comparison with field test results) is shown in Fig. 5. Note the similar trend in the data curves as that of the simulated data shown in Fig. 4. The interesting point of note regards the case of using Lot ID as the 4-NN classification parameter. In this case there are 1,350 separate classes of lots that reside in the dataset. To achieve 90% performance of the system it is only required that approximately 30% of the original DB be maintained for indexing and retrieval. The result is either a broader representation of the manufacturing data in the DB for the same available capacity, or a reduction in the amount of capacity that must be maintained in the DB system. Regardless of the result, this data management approach represents the first available means to the semiconductor industry to estimate the redundancy in the continuous stream of data being captured by the fab DMS and impose meaningful reductions based on visual similarity of images.

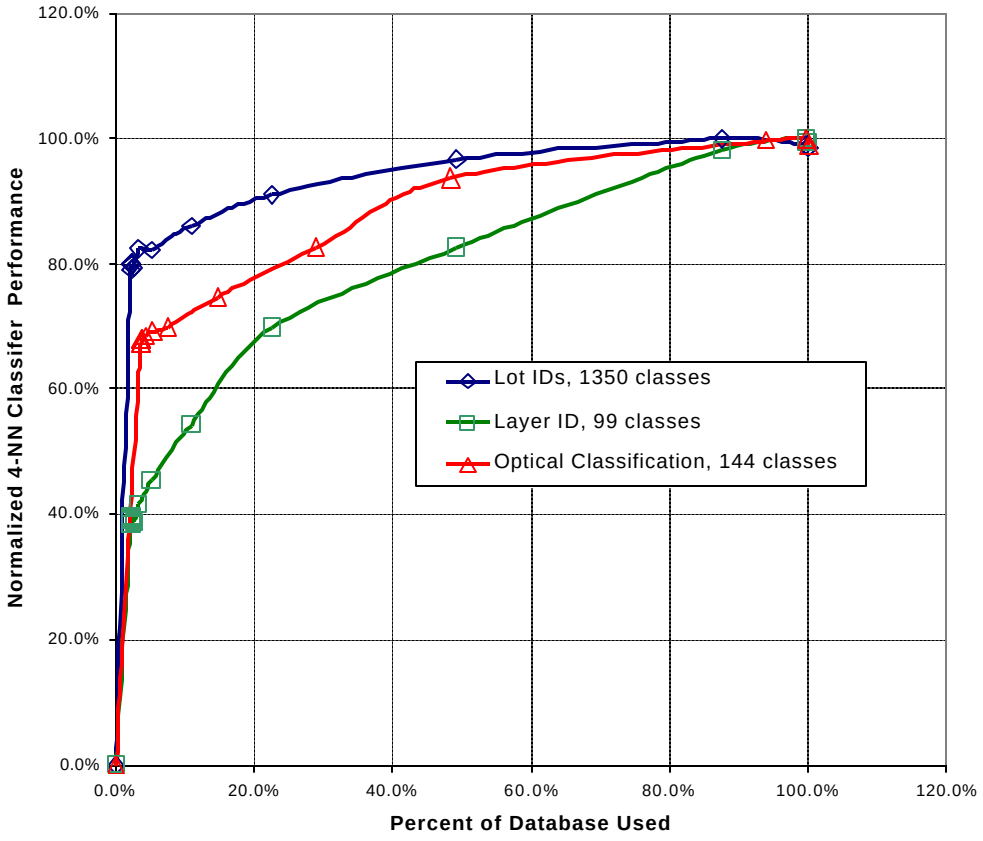

Figure 5 – Example showing data management results for actual fabrication data collected during field-testing. Note the similar trend in the reduction of the number of data points required to adequately represent the manufacturing process.

6. CONCLUSION

This work did not explore all the possible ways that AIR could be used to manage and select images for retention. As mentioned in the paper, there are many possible methods for computing the similarity value and the decision for adding or rejecting an image. Indeed, a more complete case should also address additional image aspects such as the lot number, any maintenance information associated with the image, etc. However, this work does illustrate our basic premise: the AIR system can be used to remove redundancy in an image database and reduce the size of the image database without hampering its ability to achieve defect identification goals. In the case of our experimental data, we found that the database could be reduced to 30% of its intended size and achieve 90% of the same goals as the complete system.

Further work in this area includes more elaborate functionality including the image similarity, non-image based similarity (e.g., integrating results of queries on lot IDs, layers, etc.), and decision function structures. In addition, it would be desirable to conduct further testing to validate the ideas and concepts of IDM as extended to real-life yield analysis.

7. ACKNOWLEDGEMENTS

We would like to thank International SEMATECH and its member companies for support in this project as well as data for the experimental results.

8. REFERENCES

1. Tobin, K.W., Karnowski, T.P., Lakhani, F., "The Use of Historical Defect Imagery for Yield Learning", The 11th Annual IEEE/SEMI Advanced Semiconductor Manufacturing Conference and Workshop, Fairmont Copley Plaza Hotel, Boston, MA, September 12-14, 2000.
2. Karnowski, T.P., Tobin, K.W., Ferrell, R.K., and Lakhani, F., "Content Based Image Retrieval for Semiconductor Manufacturing", SPIE's 12th International Symposium on Electronic Imaging: Science and Technology, San Jose Convention Center, January 2000.
3. Karnowski, T.P., Tobin, K.W., Arrowood, L.F., Ferrell, R.K., Goddard, J.S., and Lakhani, F., "Field-test results of an image retrieval system for semiconductor yield learning", SPIE's 13th International Symposium on Electronic Imaging: Science and Technology, San Jose Convention Center, January 2001.
4. S. Arya, D.M. Mount, N. S. Netanyahu, R. Silverman, and A.Y. Wu, "An Optimal Algorithm for Approximate Nearest Neighbor Searching in Fixed Dimensions", Proc. of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, 1994, pp. 573-582.