IMTC 2004 - Instrumentation and Measurement
Technology Conference
Como, Italy, 18–20 May, 2004

Image-Based Navigation on a Chip

M. Lifshits, E. Rivlin, M. Rudzsky

Department of Computer Science,
Technion Israel Institute of Technology,
Haifa 32000, Israel
Phone: +972 4 8293864, Fax: +972 4 8293900, E–mail: protezhe@cs.technion.ac.il.

Abstract *– In semiconductor industry, where highest levels of precision and robustness are required, machine vision tools evolved to become a mainstream automation tools that guide robotic handling, assembly and inspection processes. This paper presents a new algorithm for navigation on a chip that is based on localization of microscopic eye-point images using a previously acquired wafer map. It is fast enough for in-line microscopy and robust to visual changes occurring during the manufacturing process, such as contrast variation, re-scaling, rotation and partial feature obliteration. The algorithm uses geometric hashing, a highly efficient technique drawn from the object recognition field. Experimental results indicate high reliability of the algorithm.*

Keywords *– Integrated circuits manufacture, Integrated circuits measurements, Pattern matching, Image matching, Navigation.*

I. INTRODUCTION

Microscopic imaging is used in most core technology processes where integrated circuit (IC) digital images reveal important information. Due to the high value and enormous quantity of wafers being manufactured, the need for rapid, non-destructive imaging and measurement is growing [1], [2]. Usually, imaging is the first, but can also be the only step in the "see - measure - control" sequence in semiconductor manufacturing. Development of innovative metrology methods, designed to give faster, more detailed and useful information, which would help establish more automated process control, is required.

In this paper we refer to the act of directing a tool from one point on a chip to another as "navigation on a chip". Various types of semiconductor manufacturing equipment call for rapid and precise navigation on a chip / wafer, so that operations such as lithography, cutting and inspection can be performed to extremely tight tolerances. For example, the speed and accuracy of wafer steppers, opto-mechanical machines that perform the exposure part of the photolithography, are largely determined by the performance of a wafer stage - the part of the wafer stepper that positions a wafer under the lens-system. That is why the problem of navigation on a chip became of major importance.

Let us consider any metrology/inspection application scenario as an example of navigation on a chip. The required measurement or inspection is performed on a specific features at known positions on a wafer. In order to perform a metrology or inspection step one should first arrive at that feature. Without feedback, the stage cannot identify whether it has arrived at the exact location, and therefore, may miss the feature due to less than ideal environmental conditions such as vibrations, which contribute to stage inaccuracies. Moreover, if multiple movements are made, position errors will continue to accumulate. Modern metrology and inspection applications require highly accurate positioning of the feature to be measured in the field of view of the tool. Thus, dead-reckoning is not sufficient for navigation, and sensory feedback is required.

Current metrology techniques adopt the following solution. An additional, large and easier to track feature is selected as an "acquisition target" near the desired metrology/inspection feature. First, the nearby area is searched for the acquisition target using a low magnification level; then the location of the feature to be measured/inspected is detected by knowing its relative position. The measurement or inspection is performed using highest magnification available. This process is time consuming and non-robust as the correlation based matching techniques used for feature localization are highly sensitive to possible changes in the visual appearance of the feature. Such changes occur during the manufacturing process and may include contrast variation, re-scaling, rotations and partial feature obliteration [3].

This work proposes a new navigation algorithm, which utilizes image sensory feedback and improve upon traditional slow, non-fault-tolerant techniques, which require predefined landmarks, to be specially placed on the wafer. The algorithm exhibits high tolerance to run-time process variations and possible illumination changes.

It is fast enough for inline microscopy and robust to illumination changes and process variations. The same methodology can be used to identify known defects and perform many other tasks, essential in semiconductor manufacturing. It proved to be highly reliable when tested on typical wafer images.

II. PROPOSED APPROACH FOR NAVIGATION

Our approach for navigation on a chip is based on localization of microscopic eye-point images using a previously acquired wafer map. The algorithm uses advanced image processing techniques and geometric hashing [4], [5], [6], [8], a highly efficient technique drawn from the object recognition field.

A. Navigation Problem as Object Recognition Task

Object recognition is a known problem in the computer vision field. Recognition is achieved by finding the correspondence between a given object and a set of predefined objects. In the model-based object recognition approach, the descriptions of previously known objects are prepared in terms of various properties, such as shape, color, etc. These descriptions are referred to as "models". A given query object is then matched to one of these models.

We refer to a partial image of a chip (e.g. the current field of view of the microscope imaging system) as the wafer "eye-point". Localization on a chip is defined in the following manner: given an eye-point of the wafer, determine its exact coordinates on the wafer map. Thus, the process of navigation on a chip can be carried out by repetitive vision-based localizations. Accordingly, map-based navigation can be interpreted as model-based object recognition as follows. Suppose that we already have a description of the wafer structure - a wafer map (the process of its construction will be explained later in this section). The wafer map can be divided into many adjacent parts to be identified during navigation. These correspond to a model set in the object recognition framework and the eye-point plays the role of a query object. Matching the current eye-point to one of the previously constructed parts of the wafer map during navigation is essentially the same as associating a query object to one of the known models in object recognition. The example of the wafer eye-point and corresponding part of the wafer map are shown in Fig.1. Various approaches were applied to vision-based localization. For extensive survey of the developments in this area we refer the reader to [7]. In our case, in order to handle the enormous amounts of geometric structures contained in wafer images, we choose to address the localization problem using an efficient technique from object recognition field, called geometric hashing [4], [5], [6], [8]. Matching between a current eye-point and the wafer map is achieved by spatial correspondence of geometric features extracted from the images. The technique successfully deals with various visual transformations observable in semiconductor manufacturing such as 2D rotations, translations and uniform scale.

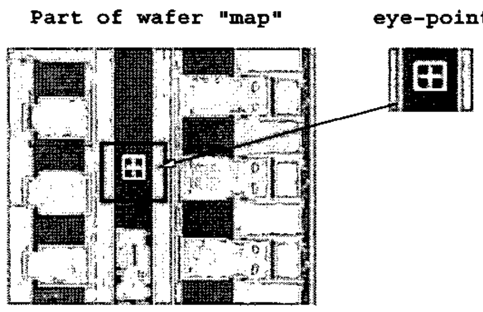


Fig. 1. Example of the real eye-point within a wafer map.

Before one can begin navigation on a chip, he or she must construct the wafer map. We construct the map from the combination of multiple partial images of the wafer lying near each other. Small images are captured by a microscope imaging system moving above the wafer surface according to some grid structure and then integrated into one big map. The process is similar to tiling the whole map with small square tiles (images). The wafer map can also be constructed from its layout file describing the two-dimensional shapes on the different layers of a chip by a limited set of graphics primitives.

After a large scale wafer map is constructed, it is decomposed into many models, which will be recognized during navigation. As a final step of the preprocessing, the representations of each model (or small partial image) are inserted in a database. When analyzing a current eye-point during navigation, the database is searched for the most suitable model among all the precompiled ones. The map coordinates of the matched model are well known.

B. The Navigation Algorithm

Our algorithm uses geometric hashing, introduced by Lamdan and Wolfson [8] which is based on the indexing approach described in [9]. The algorithm solves the problem of navigation by applying object recognition in the following way. We assume a set of predefined geometric models $M_1, ..., M_n$, defining a wafer map, and a query eye-point image Q, formed from one of the models. The task is to find the corresponding model M_i given the query eye-point Q.

It is assumed that the models are defined by a set of geometric features (e.g. corner points) and that the same features can be extracted from the query eye-point image. A model can undergo similarity transformations to form the eye-point: it can be rotated, translated and uniformly scaled. One way to make feature points invariant under this class of transformations is to represent them in the coordinate frame formed from the points themselves. For example, we may arbitrarily choose an ordered pair of model points to form a basis and describe the rest of the features in this coordinate frame. As there are multiple ways to choose a basis, we are faced with a combinatorial problem of finding the right one to match a model to the eye-point.

The algorithm copes with this problem by shifting the computational burden to the off-line learning stage. Instead of going over all feasible eye-point/model bases couples and trying to match them, all possible model representations are prepared in advance and stored in a hash table for efficient access. Thus, a query eye-point projected onto an arbitrarily chosen basis has a matching model representation already stored in the hash table.

Let $\{\mathbf{q}_1, \ldots, \mathbf{q}_k\}$ be the feature points of the eye-point Q and $M_i' = \{\mathbf{m}_1, \ldots, \mathbf{m}_k\} \subseteq M_i$ be the corresponding feature points of the matching model M_i stored in the hash table. Let us denote the transformation model M_i undergoes to form the eye-point by T; then $Q = \mathrm{T}(M_i')$, or $\mathbf{q}_j = \mathrm{T}\mathbf{m}_j$ for $1 \leq j \leq k$. Consider an ordered points pair $(\mathbf{m}_1, \mathbf{m}_2)$ of M_i'. A vector $(\mathbf{m}_1 - \mathbf{m}_2)$ with another vector rotated by 90° form the basis of a 2D coordinate frame. The coordinates (α, β) of any other point $m_j \in M_i'$ for $3 \leq j \leq k$, in this frame agree with:

$$\mathbf{m}_j = \frac{\mathbf{m}_1 + \mathbf{m}_2}{2} + \alpha(\mathbf{m}_2 - \mathbf{m}_1) + \beta(\mathbf{m} - \mathbf{m}_1),$$

where we denote an end point of the rotated vector $\mathrm{Rot}_{90}(\mathbf{m}_1 - \mathbf{m}_2)$ by $\mathbf{m}$. These (α, β) coordinates will remain unchanged when any linear transformation is applied to the model points. Application of a linear transformation T on the model M_i transforms the point $\mathbf{m}_j$ to:

$$\mathrm{T}\mathbf{m}_j = \frac{\mathrm{T}\mathbf{m}_1 + \mathrm{T}\mathbf{m}_2}{2} + \alpha(\mathrm{T}\mathbf{m}_2 - \mathrm{T}\mathbf{m}_1) + \beta(\mathrm{T}\mathbf{m} - \mathrm{T}\mathbf{m}_2),$$

so the point Tm_j has the same coordinates (α, β) in the frame formed by the ordered basis pair $(\mathrm{T}\mathbf{m}_1, \mathrm{T}\mathbf{m}_2)$. Thus, we further refer to coordinates (α, β) as invariant coordinates.

Assuming that the model M_i contains N_i feature points, there are $\binom{N_i}{2}$ different bases for that model. To form a transformation-invariant model representation, the invariant coordinates (α, β) are computed using each one of these bases $B_{\mu\nu} = \{\mathbf{m}_\mu, \mathbf{m}_\nu\}$ for every other model point. The corresponding entry $(M_i, B_{\mu\nu})$ is stored in the hash table with index (α, β).

For example, consider the left side of Fig. 2. An invariant representation of a chair model comprising five dots is inserted into the database. An eye-point containing the chair was rotated and scaled (in Fig. 2, middle). During navigation, when a pair of dots $\mathbf{p}_2, \mathbf{p}_5$ is used as the basis of a reference frame in the eye-point (Fig. 2, right side), the entry ("Chair", $\{\mathbf{p}_2, \mathbf{p}_5\}$) is repeatedly addressed in the hash table at indices $(\alpha j, \beta j)$, corresponding to the invariant coordinates of all the other points. As a result, this entry gets a sufficient number of votes and the eye-point is correctly matched to the model of the chair.

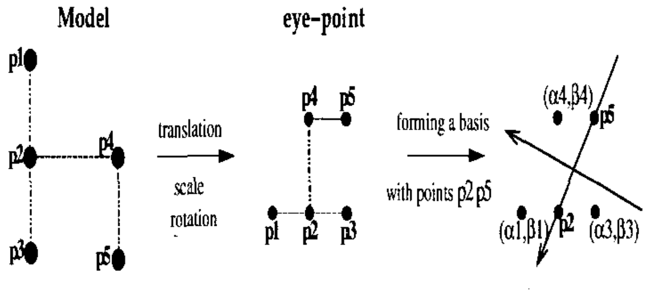


Fig. 2. Determining the hash table entries when points 2 and 5 define a basis. The models are allowed to be rotated, translated and scaled.

Navigation on a chip involves two stages: the *off-line preprocessing stage*, when most of the time consuming work to compute the invariant model representations is done, and the fast *on-line navigation stage*, which uses the data prepared by the first stage to perform the matching and navigation. The efficiency of the second stage determines the actual navigation time. We now provide detailed description of the stages.

- **Preprocessing Stage**
  In the preprocessing stage all models are processed individually, while their transformation-invariant representations are constructed and stored in a hash look-up table. For each model M and for every feasible basis b (composed of ordered features pair), it is necessary:
  1. To compute the invariant coordinates of all the remaining feature points in terms of the basis b;
  2. To use the computed coordinates as an index to the hash table and to store an entry $(M; b)$;

  The complexity of this stage is $O(n^3)$ per model, where n is the number of points contained in the model. However, since this stage is executed off-line, its complexity is of little significance.
- **Navigation Stage**
  In the navigation stage the invariant coordinates are computed from the eye-point features and are used as indexing keys to access the hash table and vote for the possible model matches. The model that scores the highest number of votes indicates the correspondence of the current eye-point with that model. An example of a typical navigation process is presented in Fig. 3. Given an eye-point with k feature points, the following steps are performed:
  1. Choose a feasible pair of feature points as a basis b;
  2. Compute the invariant coordinates of all the remaining feature points in terms of this basis b;
  3. Use each computed invariant coordinate to index into the hash table and vote for all (M_i, b_j)'s retrieved from this bin;
  4. Build a histogram for all (M_i, b_j)'s according to the number of received votes;
  5. Establish a hypothesis of correspondence between an eye-point and an instance of model M_i if (M_i, b_j), for some j peaks in the histogram with a sufficient number of votes;
  6. Verify all the hypotheses established in Step 5 and repeat from Step 1 if all of them fail verification.

  The complexity of this stage is $O(n) + O(t)$ per probe, where n is the number of points contained in the eye-point and t is the complexity of verifying one model. Note that the complexity is independent of the number of models stored in the system, thereby allowing fast navigation even on very large scale maps.

III. VERIFICATION

The second stage of the navigation algorithm is completed by verification. Given a set of candidate models that accumu-

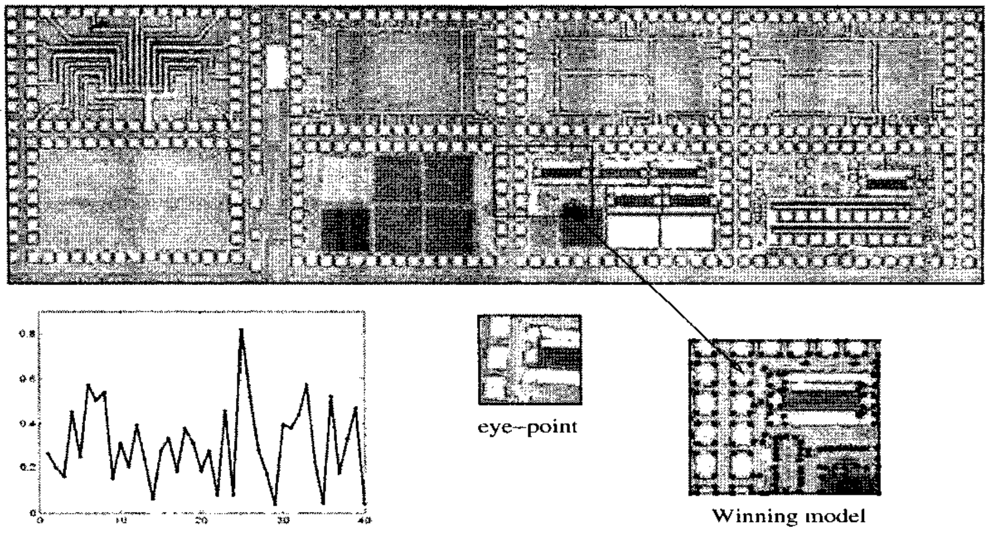


Fig. 3. Outline of the navigation process. A wafer map that is constructed from 40 model images is shown on the top. Voting results for the eye-point shown in the middle are plotted on the left. The enlarged image of the winning model (25) with its feature points marked with black dots is presented on the right.

lated the highest number of votes, one has to determine which is the best match to the query eye-point. For this purpose it is essential to specify how to fit the candidate models to the eye-point.

A. Approximating the best solution

To form the eye-point, the models undergo similarity transformation, which is a composition of translation, rotation and isotropic scaling. Thus, fitting a model to an eye-point should be done by a similarity transformation estimation. The eye-point is characterized in terms of a feature points set $\{\mathbf{x}'_i\}$ in $\mathbb{P}^2$, and each of the candidate matching models is described by its feature points $\{\mathbf{x}_i\}$, where $\mathbb{P}^2$ is a projective space. First, it is essential to find all $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ point correspondences so as to compute a similarity transformation $\mathrm{H_S}$, which transforms a model to the eye-point: $\mathrm{H_S}\mathbf{x}_i = \mathbf{x}'_i$ for each i. Two correspondences are enough to fully constrain $\mathrm{H_S}$, as the total number of degrees of freedom for similarity is four (one for the rotation, two for the translation and one more for scaling) and every correspondence gives rise to two independent equations in the entries of $\mathrm{H_S}$. However, since the locations of points in the query eye-point are not exact (due to noise), all of the correspondences should be used to determine the "best" transformation, given the data. Accordingly, $\mathrm{H_S}$ is calculated by finding the least-squares solution of the over-determined linear system.

B. Robust and efficient detection of a maximum point correspondences set

An important issue is how to efficiently find all of the correspondences. The navigation voting stage provides one corresponding basis (two point-to-point correspondences) between the candidate model and the eye-point. This allows us to approximate the desired transformation $\mathrm{H_S}$ by $\widehat{\mathrm{H_S}}$ and then, after applying $\widehat{\mathrm{H_S}}$ on the candidate model, every model point $\widehat{\mathrm{H_S}}\mathbf{x}_i$ will correspond to the closest eye-point feature $\mathbf{x}'_i$. Formally,

$$\mathbf{x}'_i = \arg\min_k d(\mathbf{x}'_k, \widehat{\mathrm{H_S}}\mathbf{x}_i)$$

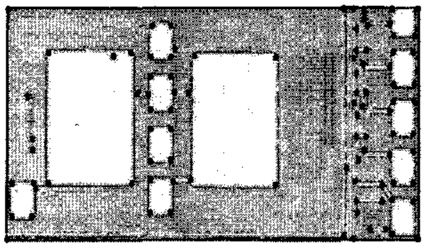
where subindex k indicates any eye-point feature and $d(\mathbf{x}, \mathbf{y})$ is the Euclidian distance between two points $\mathbf{x}$ and $\mathbf{y}$.

Thus, to compute all of the point correspondences it is possible to check the distance of each point $\mathbf{x}'_i$ to every transformed model point $\widehat{\mathrm{H}}\mathbf{x}_i$. If the model contains m points and the eye-point contains n points, those inter-set distances are computed in $O(m\,n)$ time.

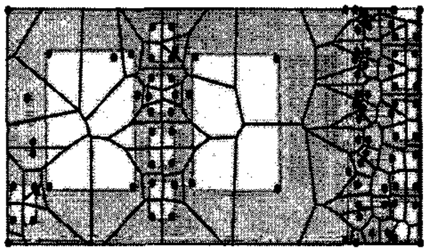
This computation can be accelerated by employing a *Voronoi tessellation* [10] for segmentation of the eye-point image. Voronoi tessellation is partitioning of a plane with n points into n convex polygons such that each polygon contains exactly one point and every point in a given polygon is closer to its central point than to any other.

We start the verification by constructing the Voronoi tessellation from the points in the query eye-point, which is done in $O(n\log(n))$ time [10] (see Fig. 4). This allows us to find the corresponding point of $\mathbf{x}_i$ in $O(log(n))$ by checking what polygon within the Voronoi tessellation contains the transformed point $\widehat{\mathrm{H}}\mathbf{x}_i$ and choosing its center point. It follows that the time needed for point correspondences calculation is reduced from $O(m\,n)$ to $O(m\,log(n))$.

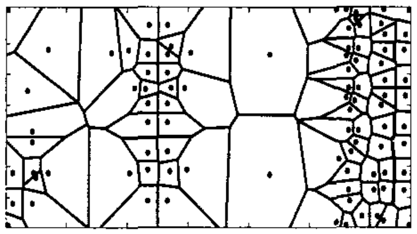
In practice the situation is complicated by the fact that some eye-point feature $\mathbf{x}'_i$ might be mistakenly reported and will not match any model point. The mismatched points, outliers, can severely disturb the estimated transformation, and consequently should be identified. In order to make the verification



(a) Eye-point image with feature points



(b) Voronoi tessellation of the eye-point image



(c) Voronoi diagram of the eye-point image

Fig. 4. The process of constructing the Voronoi tessellation of the eye-point for verification acceleration.

robust to outliers, one has to obtain a big enough set of inliers from the presented correspondences so that the transformation can be re-estimated in an optimal manner (solving the over-determined linear system as previously explained).

For this propose a general robust estimator, the RANSAC algorithm [11], is used. It is able to cope with a large proportion of outliers and has the important property of reporting an explicit number of inliers, which is important to intuitively reject false positive candidates. Given an initial putative set of correspondences, $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, computed from the model to the eye-point primary basis correspondence, the algorithm performs the following steps. A sample of two corresponding points is randomly selected and the appropriate approximation of the transformation $\widehat{\mathrm{H}}$ is calculated. Then, the number of inliers consistent with $\widehat{\mathrm{H}}$ is computed in terms of correspondences for which $d(\widehat{\mathbf{x}}'_i, \mathbf{x}'_i) < t$, where $\widehat{\mathbf{x}}'_i = \widehat{\mathrm{H}}\mathbf{x}$. These steps are repeated several times and the $\widehat{\mathrm{H}}$ with the largest number of inliers is chosen (in case of a tie the solution having the minimal standard deviation of inliers is chosen). We would like to choose a threshold t, such that with a probability α, the point is an inlier. It is assumed that the position error is Gaussian with zero mean and standard deviation σ. Thus, the square of the point distance, d^2, is the sum of the squared Gaussian variables and follows a chi-squared χ^2 distribution with two degrees of freedom (as d^2 is the sum of squared x and y measurement errors). The cumulative chi-squared distribution $F(k)$ represents the probability that the value of the χ^2 random variable is less than k. Hence, to guarantee with probability α that the point is an inlier, one has to choose a distance threshold $t^2 = F^{-1}(\alpha)\sigma^2$. Some values of t associated with different levels of certainty that the point is an inlier are tabulated in Table I. For example, if one wants to make sure that an inlier is incorrectly rejected only 5% of the time, he or she will choose α as 0.95 and accordingly $t^2 = 5.99\sigma^2$.

α	0.9	0.92	0.95	0.99
t^2	$4.6\,\sigma^2$	$5.05\,\sigma^2$	$5.99\,\sigma^2$	$9.21\,\sigma^2$

TABLE I

VALUES OF DISTANCE THRESHOLD t FOR WHICH THE PROBABILITY THAT THE POINT IS AN INLIER EQUALS α.

IV. EXPERIMENTS

In this part of the article we show the capabilities of the proposed navigation algorithm and provide an evaluation of its performance and efficiency. We performed tests on real wafer map covering an area of 4.5mm $\times$ 9mm on the wafer surface. We used adjacent microscopic images obtained on KLA-Tencor 5200XP overlay metrology tool (at 750 micron field of view) to create the wafer map. The examples of the images can be seen in Fig. 5. In order to index the hash table during navigation and vote for the correct model an eye point invariant description is calculated. This description is based on a pair of features which form a basis, as illustrated in Section II B. In practice it is possible that one of the points used to

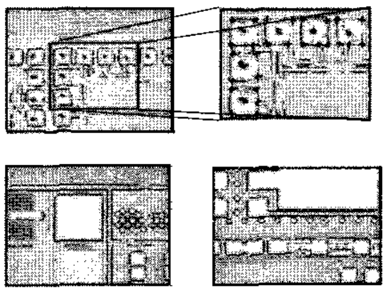


Fig. 5. Examples of the wafer images used as models in the algorithm.

form a basis will be reported by mistake and, as a result, not match any model point. In order to avoid this, many multiple attempts should be made using different bases (e.g. different descriptions), to ensure with sufficiently high probability that at least one of them is free of outliers.

In order to evaluate the navigation algorithm we varied the number of different eye-point feature bases being used in voting. To obtain a statistically meaningful measure of the algorithm performance we tested it on a total of 10^4 different navigation tasks. Each time we selected a random eye-point and then, if the correct location on the wafer "map" was reported by the algorithm (ground truth was available due to the nature of data set formation), the result was considered to be true positive (TP). The result was considered a false positive (FP) or a miss if an incorrect or no location was detected, correspondingly.

The summary of the obtained results is presented in Fig. 6. The hit rate HR $= \frac{\sharp\text{TP}}{\sharp\text{Tests}}$ reaches 95% with a 4% false alarm rate and a 1% miss rate when four bases are used.

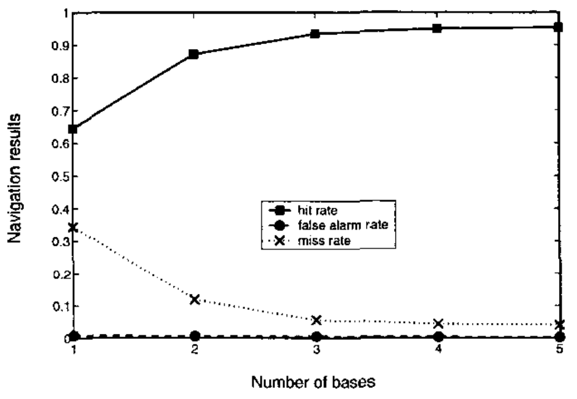


Fig. 6. System behavior with a different number of bases used in voting.

We have formulated the accuracy of the navigation result as follows. Assuming the eye-point features are measured with a Gaussian error of standard deviation σ, it can be shown that [12]: the root mean square (RMS) distance of the estimated point location from its true value is: $\sigma(2/n)^{1/2}$, where n is number of correspondences used. Thereby, for 50 sample points the algorithm estimation error is 0.2 pixels, taking $\sigma = 1$. As a result if an eye-point image of size 200×200 pixels is taken at resolution of 50 micron, our algorithm provides navigational accuracy of 0.05 μm.

As the constructed map contains areas difficult for navigation such as areas without any distinguishable features or features filled with repetitive geometric structures, we did not achieve 100% HR. However, a human would most probably also encounter serious difficulties in solving the task of self-localization for degenerate eye-points selected from these unfavorable areas. In general, our algorithm performs well for eye-points from most of the wafer areas.

V. CONCLUSIONS

We presented a new method for navigation on a chip, based on the geometric hashing technique. We showed how verification can be significantly accelerated by applying a Voronoi tessellation of the eye-point. Experimental analysis demonstrates the high reliability of the proposed method. The same principles can be used to identify known defects (by efficiently searching a defects database for the corresponding defect model) and to perform many other tasks essential for semiconductor manufacturing.

ACKNOWLEDGEMENT

This work was conducted as part of the Wafer Fab Cluster Management (WFCM) Consortium supported by the "MAGNET" program of the Chief Scientist Office at the Israeli Ministry of Industry and Trade. The authors would like to thank KLA-Tencor Israel for a donation of the 5200XP metrology tool and for support in installation and training.

REFERENCES

[1] International technology roadmap for semiconductors 2001. Semiconductor Industry Association. [Online]. Available: http://public.itrs.net/Files/2001ITRS/Home.htm
[2] International technology roadmap for semiconductors 2002. Semiconductor Industry Association. [Online]. Available: http://public.itrs.net/Files/2002Update/Home.pdf
[3] S. Melikian, Geometric searching improves machine vision, *Lasers and Optronics*, vol. 18, no. 7, p. 13, July 1999.
[4] A. Kalvin, E. Schonberg, J. T. Schwartz, and M. Sharir, Two-dimensional, model-based, boundary matching using footprints, *Int. Journal of Robotics Research*, vol. 5, no. 4, pp. 38-55, 1986.
[5] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, Affine invariant model-based object recognition, *IEEE Trans. Robotics and Automation*, vol. 6, no. 5, pp. 578589, October 1990.
[6] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, Object recognition by affine invariant matching, Proc. *IEEE Conf. on Computer Vision and Pattern Recognition*, Ann Arbor, MI, USA, pp. 335344, June 1988.
[7] G. N. DeSouza and A. C. Kak, Vision for mobile robot navigation: A survey, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 237267, February 2002.
[8] Y. Lamdan and H. J. Wolfson, Geometric hashing: a general and efficient model-based recognition scheme, Proc. *IEEE Int. Conf. on Computer Vision*, Tampa, FL, USA, pp. 238249, June 1988.
[9] J. Schwartz and M. Sharir, Identification of paritially obscured objects in two and three dimensions by matching noisy characteristic curves, *Int. Journal of Robotics Research*, vol. 6, no. 2, pp. 2044, 1986.
[10] M. V. Kreveld, M. Overmars, O. Schwarzkopf, and M. V. K. Mark de Berg, *Computational Geometry*, 2nd ed. Berlin: Springer Verlag, 2000, ch. Voronoi Diagrams: The Post Office Problem, pp. 147163.
[11] M. A. Fischler and R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Commun. ACM*, vol. 24, no. 6, pp. 381395, 1981.
[12] R. I. Hartley and A. Zisserman *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.