2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Automated Nanomanipulation with Atomic Force Microscopes

Babak Mokaberi, *Student Member, IEEE,* Jaehong Yun, Michael Wang, and
Aristides A. G. Requicha*, Life Fellow, IEEE*

Abstract—**Automation has long been recognized as an important goal in AFM (Atomic Force Microscope) nanomanipulation research. For the precise manipulation of small particles with sizes on the order of 10 nm, however, automation has remained an elusive goal, primarily because of the spatial uncertainties associated with the positioning mechanisms of the AFM and with the manipulation process itself. Extensive user intervention has been necessary for the construction of desired nanostructures with the AFM, resulting in very low throughput, and severely limiting the complexity of structures that could be built with a reasonable amount of time and labor. This paper describes a fully automatic system for building arbitrary planar patterns of nanoparticles by AFM manipulation. Given an initial, random distribution of particles on a substrate surface and a desired pattern to be formed with them, a planner determines the paths required to perform the manipulation operations. The output of the planner is a sequence of primitive commands for positioning and pushing operations involving motion along line segments. The primitive commands are executed through software that compensates for thermal drift, creep and hysteresis. Experimental results presented here show that the system can build in minutes a pattern that normally would take an experienced user a whole day to construct interactively.**

Index Terms—**Atomic Force Microscopes; AFMs; automatic nanomanipulation; drift; creep; hysteresis; nanorobotics.**

I. Introduction

Atomic force microscopes (AFMs) have been widely used in the past decade for the assembly and manipulation of nanoscale objects—see e.g. [1] and [2] and references therein. Nanomanipulation has been utilized in such applications as plasmonic device prototyping [3], building of templates such as stamps or molds, prototyping of single electron transistors [4], and manipulation of biological molecules [5]. In principle, AFM nanomanipulation can be very accurate and deal with very small objects, but its low throughput precludes the assembly of a large number of nanoobjects. Throughput can be improved by parallelism (using multi-tip arrays) or by automating the manipulation process, thus bypassing the time-consuming and labor-intensive interactive process that is typically used today to manipulate objects with overall dimensions on the order of 10 nm or less.

Manipulation of nanoparticles is routinely performed today with AFMs in ambient temperature and in air or liquids [6]. Bottom-up assembly of structures with building blocks as small as a few nanometers can be achieved by precise positioning of nanoparticles into a desired pattern with the AFM. The precision and the speed of nanomanipulation are two important factors in the construction of a dimensionally well-defined pattern in a minimum amount of time. In the past, it has been impossible to achieve speed and precision simultaneously. Precise positioning required a user in the loop, with extensive intervention to compensate for the many spatial uncertainties associated with AFMs and the nanomanipulation process. Uncertainties are introduced by phenomena that range from non-linearities in the voltage-displacement curves that characterize the AFM piezo drives, to creep, hysteresis [7], and thermal drift. The latter is the major cause of spatial uncertainty in our lab, and is due primarily to thermal expansion and contraction of the AFM components [8]. Compensation for nonlinearities in the piezo scanner is needed for achieving precise nanomanipulation with accuracy on the order of ~1nm, and can be accomplished either by closed-loop feedback [9] or a feedforward method [7].

An AFM motion in the horizontal plane of the sample is commanded by applying voltages to the instrument's piezoelectric drive motors. However, the actual displacement of the piezos and the tip positions cannot be calculated solely from the applied voltages, because of time-dependent and nonlinear effects such as drift, creep, hysteresis and other nonlinearities inherent in the piezos and overall system. Drift compensation was addressed in [8], and in [7] a new method for the open-loop compensation of creep and hysteresis was introduced. In this paper we integrate these two compensators and address other issues that arise in robust, fast and precise automated nanomanipulation. The methods described here can be exploited in most of the commercially-available AFMs, whether or not they have piezo scanner's lateral (i.e., x and y) closed-loop feedback. In our approach all the tasks for the construction of a complicated pattern are performed in an automatic and autonomous way. These tasks involve maneuvering on the sample for finding the actual (i.e., uncertainty-compensated) positions of the randomly-dispersed particles, dealing with the non-ideal behavior of the nanoparticles during the manipulation, planning the tip paths and the sequence of manipulation operations, and avoiding possible obstacles in the paths.

The only other integrated, automated system for AFM

Manuscript received September 15, 2006. This work was supported in part by the NSF under grants EIA-98-71775 and DMI-02-09678.

B. Mokaberi , J. Yun and M. Wang are with the Laboratory for Molecular Robotics, 941 Bloom Walk, University of Southern California, Los Angeles, CA 90089-0781 (e-mail: mokaberi@usc.edu, jaehongy@usc.edu, wangmich@usc.edu).

A. A. G. Requicha is with the Laboratory for Molecular Robotics, 941 Bloom Walk, University of Southern California, Los Angeles, CA 90089-0781 (tel: 213.740.4502; fax: 213.740.7512; e-mail: requicha@usc.edu).

```
filterState = MEASUREMENT;
do forever {
  if filterState == MEASUREMENT then {
    t = getCurrentTime();
    measureDrift();
    updateDriftValue();
    updateOffsets();
    updateDriftCovariance();
    if (there is a manipulation task  AND
        covariance < lowThreshold)  then
        filterState = PREDICTION;
    else
          wait until (getCurrentTime()–t == $T_m$);
  }
  if filterState == PREDICTION then {
    t = getCurrentTime();
    predictDrift();
    updateDriftValue();
    updateOffsets();
    computePredictionCovariance();
    if (there is no manipulation task OR
        covariance > highThreshold) then
        filterState = MEASUREMENT;
    else
          wait until (getCurrentTime()–t ==$T_p$);
  }
}
```

Fig. 1. Top-level pseudo-code for the Kalman filter compensator.

nanomanipulation reported in the literature, as far as we know, is the system developed by Prof. Xi's group at Michigan State University [10]. Xi's system addresses at length the issues that arise in nanorod manipulation, and therefore is more general than ours, which focuses on nanoparticles. On the other hand, the Michigan State system has a more rudimentary drift compensator than ours, and does not compensate for creep or hysteresis, which are important for the manipulation of small objects, with dimensions ~ 10 nm or less. The manipulation tasks demonstrated in [10] involve objects which are roughly one order of magnitude larger than those we manipulate in the present paper, and the positional errors in the final structures shown in the figures of [10] are also similarly larger than ours.

The remainder of the paper is organized as follows. Section II discusses the compensation of AFM's spatial uncertainties. Section III addresses the inaccuracies associated with the manipulation procedures and explains how to combat them. Section IV introduces an algorithm for planning the tip paths for manipulation tasks. Experimental results are presented in Section V, and conclusions are drawn in a final section.

II. Compensation of AFM's Spatial Uncertainties

A. Drift Compensation

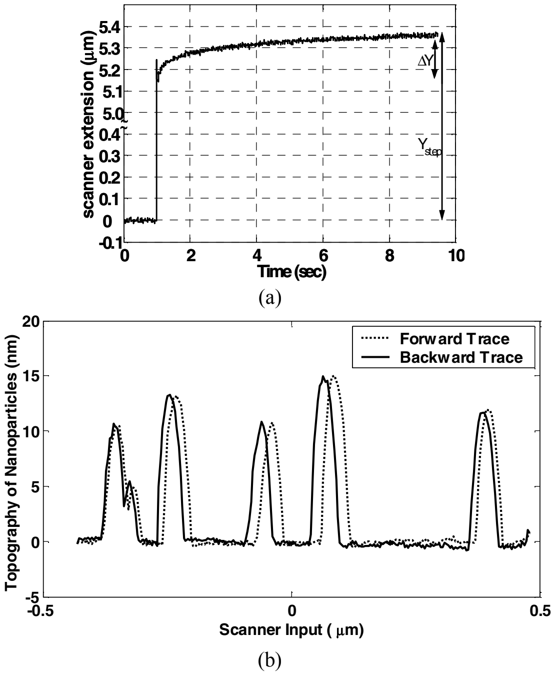
Drift arises from the mechanical and thermal expansions of AFM components. Even with the most sophisticated lateral positioning sensors used today in piezo scanners, the drift effect typically is still present during ambient-temperature operations, because in nearly all the currently-available commercial AFMs the lateral sensors do not measure the position of the tip relative to the sample (see [8]). Although the drift decreases when the instrument stabilizes, after several hours of AFM operation in the Veeco CP-R AFMs in our lab drift is still present, with rates that vary between 0.01-0.2 nm/sec. As we demonstrated in [8], a Kalman filter can effectively estimate the drift and provide a reliable prediction of the actual tip position for manipulation tasks.

Figure 1 summarizes the process of drift compensation. The translational displacement due to drift is measured either by correlating successive images in a tracking window, or by computing some geometric property of a feature being tracked, e.g., the center of a nanoparticle. The tracking window is typically small, on the order of 150x150 pixels, and can be chosen automatically to maximize its content. The filter is initially in its measurement state and acquires a series of drift measurements with a sampling interval T_m (typically T_m = 40 sec). Each measurement triggers updates of the drift values and of the covariance of the estimate by using the standard Kalman equations. If no manipulation task is scheduled or executing, the filter continuously measures the drift. When the covariance falls below a low threshold and there is a manipulation task, the filter switches to its prediction state in which the drift values and the associated covariances are calculated by the Kalman prediction equations, with a sampling interval T_p (typically T_p = 4 sec). The covariance increases with time, and when it exceeds a high threshold, the filter switches back to its measurement state. In our experience, the filter can predict drift accurately for some 15 minutes without measurements. Note that calculations in the prediction state are very fast whereas in the measurement state actual measurements and tip motions are required. Updates of drift values are propagated to the instrument's *X* and *Y* offsets, essentially causing a change of origin for the system's coordinates and compensating for the drift.

Before a positioning or manipulation task is scheduled for execution, the system checks the current values of the offsets and covariance values. If the covariance is above the high threshold, the task must wait for the filter to reduce the covariance by executing measurements. Otherwise, the task is scheduled for execution and the current offsets are added to the nominal coordinates to obtain the compensated values.

B. Creep and Hysteresis Compensator

When an abrupt change in voltage is applied to the scanner, the corresponding piezo dimensional change occurs in two stages: the first stage takes place in less than a millisecond, whereas the second one has a much longer time scale (Fig. 2a). The second stage is known as creep. Typical values for the creep ratio are from 1% to 20%, and from 10 to 100 seconds for the phenomenon's time duration.



(a)

(b)

Fig. 2. (a) The creep effect observed from the scanner response to a 5.4 μm step function. (b) Forward and backward traces over a set of 15 nm Au particles on mica. The length of the scan is 910 nm and the speed 2Hz.

Figure 2b illustrates the effect of hysteresis. A single line scan of a set of particles shows the particles in different positions, depending on whether the scan is in the forward or backward directions. Again, the effects can be large. In both Fig. 2a and Fig. 2b the motions occur within a few seconds and the effects of drift are negligible. Thus, even in a drift-compensated AFM, creep and hysteresis cause spatial uncertainties that are well beyond the threshold accuracy of a few nm that in our experiments we have found necessary for the successful manipulation of particles with sizes on the order of 10 nm.

Creep can be modeled as a linear combination of several fundamental linear operators plus a term proportional to the input, to account for sudden changes in the output. The discrete model of creep, assuming sampling period T , can be represented in the following state-space form:

$$\begin{aligned} x_i(k) &= e^{-\lambda_i T} x_i(k-1) + (1-e^{-\lambda_i T})u(k-1) \\ y_C(k) &= \sum_{i=1}^{N_c} c_i x_i(k) + au(k) \qquad c_i \geq 0 \end{aligned} \tag{1}$$

Here x_i denote the creep operator states, y_C the output, N_C the order of creep model and u is the input signal to the scanner. Hysteresis is modeled by using a similar idea, but with a nonlinear rate-independent fundamental operator called play operator [11]. The input-output behavior of a play operator with a threshold value $r > 0$ is given by

$$z_r(k) = \max\{u(k) - r, \min\{z_r(k-1), u(k) + r\}\} \tag{2}.$$

The model approximates the behavior of the hysteresis in the piezo scanner by the superposition of a sufficiently large number of play operators:

$$y_H(k) = \sum_{i=1}^{N_H} w_i z_{r_i}(k) \tag{3}$$

where N_H is the order of the hysteresis model, and $w_i > 0$ are the weights associated with the play operators.

By adding (1) and (3), the overall extension of the scanner can be defined as the sum of two terms,

$$y(k) = y_C(k) + y_H(k) = \underbrace{\sum_{i=1}^{N_c} c_i x_i(k) + au(k)}_{L[u](k)} + \underbrace{\sum_{i=1}^{N_H} w_i z_{r_i}(k)}_{P_I[u](k)} \tag{4}.$$

The first term, $L[u](k)$ is a rate-dependent term which depends on the past values of the input through the x_i terms in (1), and the second term, called a Prandtl-Ishlinskii (PI) operator, is a rate-independent term which depends on the *current* values of the input.

In [12], [13] it is proved that the PI operator is piecewise continuous and invertible, and its inverse is also a PI operator expressed as

$$P_I^{-1}[v](k) = \hat{a}v(k) + \sum_{i=1}^{N_H} \hat{w}_i z_{\hat{r}_i}(k) \tag{5}$$

with the parameters $\hat{a}$, $\hat{w}_i$ and $\hat{r}_i$ defined by

$$\begin{aligned} \hat{a} &= 1/a \\ \hat{r}_i &= \sum_{j=0}^{i} w_j(r_i - r_j) \quad i = 0,1,\ldots,N_H \\ \hat{w}_i &= \frac{-w_i}{(a + \sum_{j=1}^{i} w_j)(a + \sum_{j=1}^{i-1} w_j)} \quad i = 1,2,\ldots,N_H \end{aligned} \tag{6}.$$

In [12] it is also shown that for every continuous input signal, the PI operator is Liptschitz continuous and causally invertible as long as $a, w_i, r_i > 0$ for $i=1,2,\ldots,N_H$. This means that, under the above condition, for every continuous function v there exist a unique input function u such that

$$u(k) = P_I^{-1}[v](k) \tag{7}.$$

We always assume piecewise continuous functions for the output trajectory and use positive parameters that ensure the invertibility of PI operator. Denote the desired output trajectory of the scanner by $y_d(k)$ and replace it in the left side of (4). We can rewrite (4) as

$$y_d(k) - L[u](k) = P_I[u](k) \tag{8}$$

To solve this equation for u we let

$$v(k) = y_d(k) - L[u](k) \tag{9}$$

and apply the inverse PI operator (7) to v. Although v depends on u, we can apply the previous equations recursively to find the input u that corresponds to the desired output y_d.

In [7] we introduced a new, adaptive, recursive method for the identification of model parameters that uses solely the information from the topography of the sample and is suitable for the identification of a large number of parameters. This method is applicable to the vast majority of the AFMs currently in use, which either have no sensors on the horizontal plane of the sample, or have feedback loops that are too noisy for nanomanipulation of small nanoparticles with overall sizes on the order of 10 nm.

C. Integration of Drift, Creep and Hysteresis Compensators

Drift, creep and hysteresis need to be compensated simultaneously for accurate nanomanipulation. In certain circumstances, it is possible to reduce the effect of one or two of them by limiting the operation time or distance, but it is nearly impossible to avoid all of them at the same time.

Consider the primitive operation of moving the AFM tip from point $A(x_1, y_1)$ to point $B(x_2, y_2)$. If the covariance of the drift estimate is too high, the drift compensator is invoked, and it runs a series of measurements and updates until the covariance becomes lower than the threshold—see Fig. 1. Then the Kalman filter starts to predict the drift states as $\hat{x}(t_k^c \mid t_k)$ and $\hat{y}(t_k^c \mid t_k)$ (for *X* and *Y* directions), where t_k^c and t_k represent the current time and the time at which the last drift measurement took place. Normally the updates are computed every 4 sec and are accessible to all the AFM tasks which need a most recent estimate of drift. Thus the updated coordinates of *A* and *B* are transformed as

$$A(\hat{x}_1, \hat{y}_1):\ \hat{x}_1 = x_1 + \hat{x}(t_k^c \mid t_k)\ ,\hat{y}_1 = y_1 + \hat{y}(t_k^c \mid t_k) \qquad (10)$$

$$B(\hat{x}_2, \hat{y}_2): \hat{x}_2 = x_2 + \hat{x}(t_k^c \mid t_k), \hat{y}_2 = y_2 + \hat{y}(t_k^c \mid t_k) \qquad (11)$$

and subsequently the desired trajectory in the *x* and *y* directions is computed from

$$\begin{aligned} x_d(k) &= x_d(k-1) + \frac{T \cdot \mathbf{v}}{d(k-1)} \cdot \left(\hat{x}_2 - x_d(k-1)\right) \\ y_d(k) &= y_d(k-1) + \frac{T \cdot \mathbf{v}}{d(k-1)} \cdot \left(\hat{y}_2 - y_d(k-1)\right) \end{aligned} \qquad (12)$$

where *T* is the creep-hysteresis model sampling period, **v** is the tip's speed and $d(k-1)$ is the distance from point ($x_d(k-1), y_d(k-1)$) to the destination *B*, i.e.

$$d(k-1) = \sqrt{\left(\hat{x}_2 - x_d(k-1)\right)^2 + \left(\hat{y}_2 - y_d(k-1)\right)^2} \quad (13).$$

In (12) the initial conditions are defined as $x_d(k_0) = \hat{x}_1$, $y_d(k_0) = \hat{y}_1$. The desired trajectories are then used in the inverse model (7) and (9) for the computation of the input signal to the scanner. Each time (12) and (13) are computed for a new trajectory data point, the updated estimates of $B(\hat{x}_2, \hat{y}_2)$ from (11) are used.

III. Nanomanipulation Procedure

Manipulation of nanoparticles by mechanical pushing with an AFM tip has been discussed at length in [1], [2] and [6]. The manipulation is performed by moving along a line passing through the center of each particle and turning off the *Z* feedback. The centers of particles are automatically found by using a series of horizontal and vertical scans over the approximate positions of the particles [8]. Once the coordinates of a particle's center are accurately computed, a single line scan is performed, passing through the center and the particle's target location, and the *Z* feedback is turned off during the appropriate part of the scan. Experimental evidence shows that long range manipulations do not consistently produce accurate results–see Fig. 3. Therefore a long range manipulation is always broken into several short pieces with a length Δd typically between 30-50 nm. To move a particle from its current position *A* to its final target position *B*, we can apply the procedure in Section II.C for the computation of a manipulation path in the compensated environment. If the distance between *A* and *B* is larger than Δd, then the AFM manipulates the particle by Δd, searches for the center at the new location, and starts the manipulation again for the remaining path, using the updated estimate of the position of target *B* from (11).

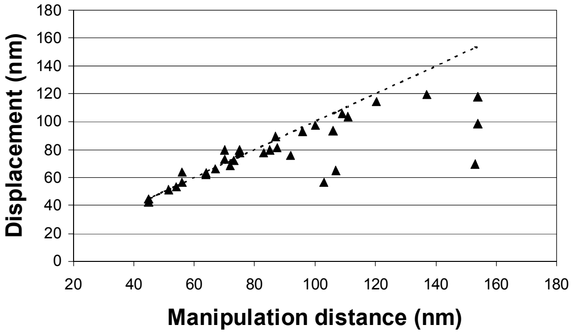


Fig. 3. The relationship between the distance that actually a 15nm particle moved and the distance initially assigned for the push. As the manipulation distance exceeds about 90nm, the position of the moved particle deviates from the desired manipulation distance.

The initial positions of the particles are estimated from a first scan of the working area. Each single horizontal scan line is flattened, and thresholded for removing background artifacts. Then the particles are labeled by using the Connected Components Labeling method [14]. The image is scanned and its pixels grouped into components based on pixel connectivity (8-connectivity in our case). All the pixels in a connected component share similar label values. After completing the scan, the equivalent label pairs are sorted into equivalence classes and a unique label is assigned to each class. As a final step, a second scan is made through the image, during which each label is replaced by the label assigned to its equivalence class. The labeled particles then can be used for the computation of centers using the center of mass method.

IV. Planning the Manipulation Path

The initial dispersion of nanoparticles on the surface is random with a density that can be roughly controlled by the colloidal deposition time [15]. Automatically planning the manipulation path to generate a desired pattern requires that each particle be uniquely assigned to a target point in the desired pattern and moved without colliding with the other particles (or possible obstacles) along the manipulation path.

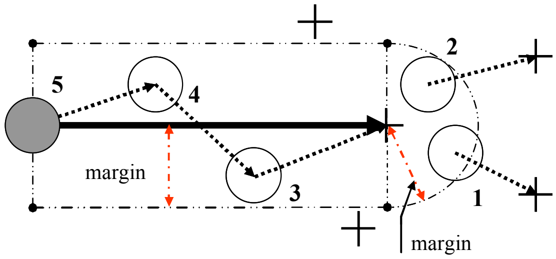


Fig. 4. Collision checking for one execution task. Numbers show the order of pushing.

In addition, the total manipulation distance should be minimized. Since a manipulation operation includes several steps of center sensing and pushing, it typically takes much longer than a simple linear movement of the tip. Therefore, reducing the total manipulation distance can drastically decrease the manipulation time and hence increase the overall throughput. We summarize below the manipulation path planning algorithm in three steps.

A. Assignment of particles to targets

Assignment of particles to destinations can be considered as a weighted bipartite graph matching problem. The weights are simply the distances from each particle to the destinations. The Hungarian algorithm [16] finds a set of direct paths between particles and target positions in $O(n^2 \cdot d)$ time, where *n* is the number of sources and *d* is the number of targets. This algorithm guarantees optimal overall manipulation distance for direct paths. If there are any other obstacles such as blocks or a line of adjacent particles, we may need more steps to find indirect paths around the obstacles.

B. Sequencing

After matching the particles to their targets, the next step is specifying an order for the manipulation operations. We assume that after manipulating a particle to its target location, the tip moves in a straight line to the initial position of the next particle in the operation sequence. Although sequencing doesn't affect the total manipulation distance, it directly influences the possibility of collisions between particles and the total movement of the tip as well. The collision problem will be addressed in the next step. Sequencing is done by a greedy algorithm. After finishing the manipulation of a particle, we compute the distances between the current tip position and the initial positions of all the particles not yet manipulated. The shortest distance determines the next particle to be manipulated. This algorithm is suboptimal. However, the time taken by the tip to move between the particles (while not pushing) is typically much smaller than the pushing time, and therefore using a better optimizer does not dramatically reduce the total task time. (Each particle is typically pushed for 5-90 seconds while the tip can move a distance of 100 nm in a fraction of a second.) We experimented with optimization algorithms such as simulated annealing, genetic algorithms or ant colony optimization and found that they could reduce the total task time by a few seconds, but they increased the computation (planning) time by more than an order of magnitude.

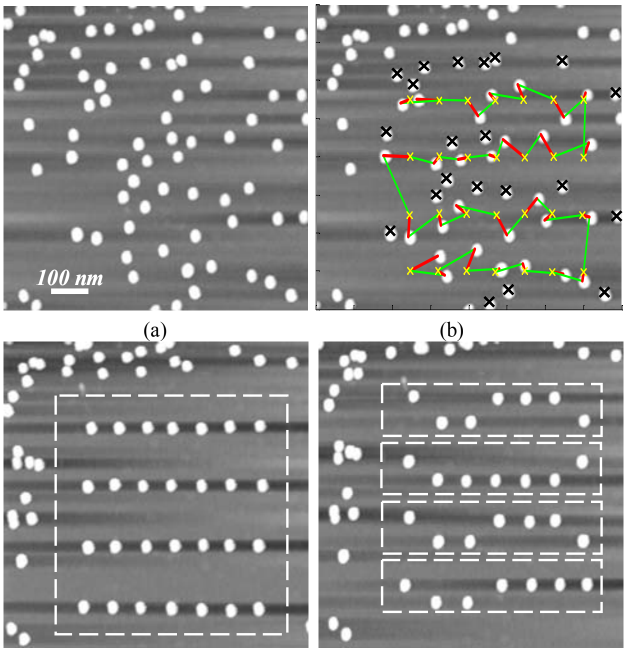
C. Collision Avoidance

In our previous work on nanomanipulation planning [17] we considered arbitrary polygonal obstacles in the plane, and used precedence constraints for sequencing and avoiding collisions. Here we take a simpler but effective approach in which we only consider other particles as potential obstacles, and use a particle substitution technique for dealing with collisions. Instead of calculating all possible cases of collisions, the planner just simulates the real pushing operations, and on each pushing task it investigates possible collisions. In checking collisions, the planner considers a margin for pushing errors by growing the pushing path by a predetermined amount (see Fig 4). In Fig. 4 the gray particle labelled 5 is to be pushed along the thick black arrow path and the four other particles shown as unfilled circles are considered obstacles. If there are collisions on the pushing path, the planner first pushes the obstacles. Thus, in Fig. 4, particles 1 and 2 are moved to their target positions. The original target position is filled by pushing the closest obstacle to it, and the position where the obstacle previously was is set as target for the next closest particle. In Fig. 4, particle 3 is moved to the target of the gray particle, and the position of 3 becomes the target of the next obstacle, which is particle 4. The simulated execution and collision checking are done recursively. In Fig. 4, particle 4 is moved to where 3 used to be, and then particle 5 is moved to the initial position of particle 4. The net result is as if the gray particle had been moved to its target position.

V. Experimental Results

The following results were obtained on the AutoProbe CP-R AFM (Veeco) with a 5 μm scanner, a sharp tip (Budgetsensors Tap300-AL) with spring constant $k \cong 25$ N/m, in dynamic mode, imaging and manipulating gold nanoparticles with nominal diameters of 15 nm, deposited on a mica surface covered with poly-L-lysine, in air, at room temperature and humidity. Filter parameters (see [8]) were selected at the nominal values of R_0=1.2 nm^2, α=0.11 min^{-1} and σ_m^2=0.048 nm^2/min^4. The orders of the creep and hysteresis models were selected as $N_C = 2$, $N_H = 8$ and the parameters were identified as (see [7]): $T = 8$ msec, $a = 0.9167$, $e^{-\lambda_i T} = [0.983,\ 0.998]$, $c_i = [0.0458,\ 0.0489]$, $r_i = [0.03,\ 0.11,\ 0.17,\ 0.23,\ 0.28,\ 0.34,\ 0.40,\ 0.46]$, $w_i = [0.033,\ 0.014,\ 0.006,\ 0.008,\ 0.006,\ 0.001,\ 0.002,$ $0.007]$. (For brevity data are given only for the *X* direction; the input and output dimensions are in micrometers.)

Figure 5 illustrates the construction of a pattern of 28 nanoparticles by using the automated manipulation procedure introduced in this paper. Figure 5a shows the initial, random dispersion of the particles on the substrate.



(a) (b)

(c) (d)

Fig. 5. Automated manipulation of 15nm Au particles on mica. (a) Initial random dispersion. (b) Output of the planner with pushing paths in red and positioning paths in green (c) Result of manipulation to form four 7-particle rows. (d) Manipulation of the particles in (c) to form the characters in ASCII code that correspond to the word “NANO” (refer to the text for details).

Fig. 5b shows the output of the planner, with the pushing paths depicted in red and the positioning paths between pushing operations in green. The particles marked with a cross are not part of the final structure and were removed interactively. (We have now enhanced the planner to be able to remove extra particles automatically.)

Fig. 5c depicts the final configuration corresponding to the plan of Fig. 5b: 4 parallel rows of 7 particles each. Fig. 5d shows the result of a second manipulation task in which several particles were moved upward and placed in new rows lying between the four original ones. A particle on a top row in each 2-row group shown in Fig 5d corresponds to a ‘1’, whereas a particle on the lower row of a group corresponds to a ‘0’. The entire pattern can be read as a set of four 7-bit ASCII characters, spelling ‘NANO’. Each bit is stored in an area about 135×82 nm^2, which corresponds to an areal density of approximately 9.0 Gbit/cm^2. Much higher densities could be achieved by using smaller particles and placing them closer together. The writing scheme used in this example was introduced in [18], which discusses at length the issues that surround the potential application of nanomanipulation to storage devices. Here we simply present these storage structures to demonstrate the operation of our automated nanomanipulation system.

The drift was measured for 10-15 minutes with a 40 sec sampling interval before starting the manipulation tasks. Before each motion in the manipulation task was executed, the covariance was checked. If it exceeded an upper threshold of 6.5 nm^2, the AFM halted its task and passed the control to the drift compensator. The compensator then measured the drift for about 3.3 minutes to bring the covariance down to ~ 0.56 nm^2. For the experiments of Fig. 5c and Fig. 5d, drift measurements were needed three and two times, respectively, while executing the manipulation tasks. The total manipulation time for the two structures together was on the order of half an hour, not including the times required for drift measurements. This is a major improvement over the results in [18], in which we built a simpler structure with 10 particles spelling ‘LMR’ in ASCII; the “LMR” structure took about one day to construct interactively by an experienced user.

VI. Conclusion

This paper describes an integrated, automated system for manipulating nanoparticles with an Atomic Force Microscope (AFM). Given an initial, random configuration of particles on a substrate surface, plus a goal pattern, a planner computes a set of collision-free paths to build the desired pattern. The planner matches the particles to target positions by using the Hungarian algorithm for bi-partite matching, which is optimal. It then sequences the pushing operations by using a suboptimal greedy algorithm, and avoids collisions with other particles through a particle substitution scheme.

The pushing and positioning paths computed by the planner are passed on to the manipulation software for execution. Drift, creep, and hysteresis are compensated in software. The drift compensator is based on Kalman filtering, and creep and hysteresis are compensated by a feedforward method based on a Prandtl-Ishlinskii model of these phenomena.

Experimental results are presented for a potential data storage application in which gold nanoparticles with diameters of 15 nm are automatically manipulated into a structure that represents a set of ASCII characters. We believe this is the first demonstration of automated nanomanipulation for particles of these small sizes. Nanomanipulation using the techniques introduced here should be able to assemble nanostructures much more complex than those constructed in the past.

References

[1] A. A. G. Requicha, "Nanorobots, NEMS and Nanoassembly", *Proc. IEEE*, vol. 91, no. 11, pp. 1922-1933, November 2003.

[2] A. A. G. Requicha, "Nanorobotics", in *Handbook of Industrial Robotics*, 2nd. ed, S. Nof, Ed., New York, NY: Wiley, 1999, pp. 199-210.

[3] S. A. Maier, P. G. Kik , H. A. Atwater , S. Meltzer, E. Harel , B. E. Koel and A. A. G. Requicha, "Local detection of electromagnetic energy transport below the diffraction limit in metal nanoparticle plasmon waveguides”, *Nature Materials*, vol. 2, no.4, pp. 229-232, April 2003.

[4] S. B. Carlsson, T. Junno, L. Montelius and L. Samuelson, “Mechanical tuning of tunnel gaps for the assembly of single-electron transistors”, *Appl. Phys. Lett.*, vol. 75, no. 10, pp 1461-1463, September 1999.

[5] J. H. Lu, “Nanomanipulation of extended single-DNA molecules on modified mica surfaces using the atomic force microscopy”, *Colloids and Surf. B: Biointerfaces*, vol. 39, no. 4, pp. 177-80, December 2004.

[6] C. Baur, A. Bugacov, B. E. Koel, A. Madhukar, N. Montoya, T. R. Ramachandran, A. A. G. Requicha, R. Resch and P. Will, “Nanopar-

ticle manipulation by mechanical pushing: underlying phenomena and real-time monitoring", *Nanotechnology*, vol. 9, no. 4, pp. 360-364, December 1998.

[7] B. Mokaberi and A. A. G. Requicha, "Compensation of Scanner Creep and Hysteresis for AFM Nanomanipulation", *to appear in IEEE Trans. on Automation Science and Engineering.*

[8] B. Mokaberi and A. A. G. Requicha, "Drift Compensation for Automatic Nanomanipulation with Scanning Probe Microscopes ", *IEEE Trans. on Automation Science and Engineering*, vol. 3, no. 3, pp. 199-207, July 2006.

[9] S. Salapaka, A. Sebastian, J.P. Cleveland and M.V. Salapaka "High bandwidth nano-positioner: A robust control approach", *Rev. Sci. Instrum.*, vol. 73, no. 9, pp 3232-3241, September 2002.

[10] H. Chen, N. Xi and G. Li, "CAD-guided automated nanoassembly using atomic force microscopy-based nanorobotics", *IEEE Trans. on Automation Science & Engineering*, vol. 3, no. 3, pp. 208-217, July 2006.

[11] I.D. Mayergoyz, *Mathematical Models of Hysteresis*. New York, NY: Springer-Verlag, 1991.

[12] P. Krejci and K. Kuhnen, "Inverse control of systems with hysteresis and creep", *IEE Proc.-Control Theory Appl.*, vol. 148, no. 3, pp. 185-192, May 2001.

[13] M. Brokate and J. Sprekles, *Hysteresis and Phase Transitions*. New York, NY: Springer-Verlag, 1996.

[14] B. Horn, *Robot Vision*, MIT Press, 1986.

[15] C. Baur, B. C. Gazen, B. Koel, T. R. Ramachandran, A. A. G. Requicha and L. Zini, "Robotic Nanomanipulation with a Scanning Probe Microscope in a Networked Computing Environment", *J. Vacuum Sc. & Tech. B*, vol. 15, no. 4, pp. 1577-1580, July/August 1997.

[16] D. E. Knuth, *The Stanford GraphBase*, New York, NY: The ACM Press, 1993.

[17] J. H. Makaliwe and A. A. G. Requicha, "Automatic planning of nanoparticle assembly tasks", *Proc. IEEE Int'l Symp. on Assembly and Task Planning, Fukuoka, Japan*, pp. 288-293, May 28-30, 2001.

[18] A. A. G. Requicha, "Massively parallel nanorobotics for lithography and data storage", *Int'l J. Robotics Research*, vol. 18, no. 3, pp. 344-350, March 1999.