Scanning probe microscopy based on reinforcement learning[a]

Eric Yue Ma[1, b]
[1]*Department of Applied Physics, Stanford University, California, US*

(Dated: 11 December 2015)

Keywords: machine learning, neural network, reinforcement learning, scanning probe microscopy

Scanning probe microscopy (SPM) has become an indispensable tool for characterizing the nanoscale. The core of its working principle involves raster-scanning a sharp tip in contact with a sample of interest, while maintaining a constant tip-sample interaction. This is achieved via standard proportional-integral (PI) feedback in virtually all industrial implementations. Here we explore 1) the possibility of using machine learning to automatically find the optimal PI parameters and 2) feedback based on a Markov decision process (MDP) reinforcement learning model, free from any explicit PI feedback. PI parameters automatically selected by a neural-network-based approach achieve excellent feedback performance. The MDP model is shown to perform basic feedback, but with inferior performance than that of PI feedback, possibly due to the intrinsic unpredictability of sample topography. These proof-of-concepts open up new opportunities for SPM feedback optimization under a wide range of conditions.

I. INTRODUCTION

Scanning probe microscopy (SPM) characterizes extremely small, often down to subatomic scale features, and has become an indispensable tool for a growing community of scientists and engineers from condensed matter physics, material science, quantum chemistry, molecular biology and semiconductor industry. In an SPM, a sharp tip is brought close to the surface of the sample of interest. The interaction between the atoms on the tip apex and those on the sample surface yields some measurable quantity (e.g. the deflection of a micro-cantilever in atomic force microscope (AFM), or the quantum tunneling current in a scanning tunneling microscope (STM)); through feedback on this quantity while moving the tip on the sample surface in a raster-scan fashion, a constant tip-sample interaction is maintained, and high spatial resolution topography images can be obtained. Many other electrical, magnetic and chemical properties can be simultaneously measured with specialized tips.

To the best knowledge of the author, all current commercially available SPMs utilize some variation of a proportional-integral (PI) feedback control scheme (the derivative (D) in a standard proportional-integral-derivative (PID) feedback scheme is usually not used due to the susceptibility to measurement noise). While the PI feedback can handle most scanning conditions given the optimal parameters, finding those parameters is often tedious and requires substantial empirical knowledge. Moreover, the optimal values can be drastically different for different tip type, sample surface roughness and even scan speed. Automatic, or semi-automatic tuning of PI parameters in an SPM setting has previously been proposed, based on the standard Ziegler-Nichols or relay method[1–3], but has not been widely available in common SPMs.

Here we aim to explore the possibility of using machine learning to improve the performance, robustness, and ease of use of an SPM. We aim to achieve two progressive goals: 1) automatic tuning of PI feedback parameters using machine learning algorithms, and 2) a MDP-based reinforcement learning model capable of pixel-by-pixel feedback, free from an explicit PI feedback scheme. Its performance will be compared to the standard PI feedback with optimal parameters. Preliminary conclusion on the feasibility, advantages and disadvantages of such MDP model will be drawn based on the results.

II. METHODS

A. SPM simulator

A simple but realistic SPM simulator is implemented in MATLAB. The state of the simulator is described by just two quantities: z_s is the height of the sample surface at the point currently underneath the tip, measured with respect to a fixed reference; z_t is the height of a fixed part of the tip, measured from the same reference. If z_s and z_t are given, the simulator calculates the tip-sample interaction signal s which only depends on the difference between z_s and z_t: $s(z_s, z_t) = s(z_t - z_s)$ (Fig. 1A, B).

The working principle of an SPM is as follows: z_s is what we want to measure but it cannot be measured directly; z_t is measurable and controllable, usually via a piezoelectric device; if $s(z_t - z_s)$ is a constant, $z_t - z_s$ is also a constant. When the tip moves across the sample surface, z_s changes in an unknown way according to $z_s(x, y)$. If we control z_t to maintain the value of s as

[a]CS229 project final report.
[b]Electronic mail: yuema@stanford.edu

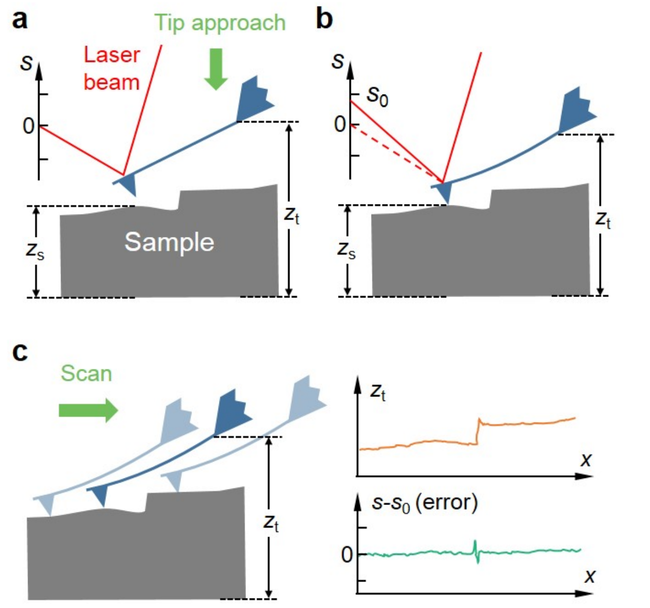

FIG. 1. **(a)** Illustration of z_s, z_t and s. In this illustration an atomic force microscopy (AFM) configuration is used, where s is the deflection of a micro-cantilever with the tip at its end, measured by the reflection of a laser beam. When the tip is not in contact with the sample, there is 0 deflection. **(b)** When the tip comes in contact with the sample surface, the cantilever is deflected, thus s increases, in this case to a predefined set point s_0. **(c)** If s is maintained at s_0 while the tip scans across the sample surface, $z_t(x)$ becomes a faithful representation of $z_s(x)$. The feedback is never perfect, and can be evaluated by the feedback error $(s(x) - s_0)$.

close to a constant s_0 (the "set point") as possible, we would have $z_t(x, y) = z_s(x, y) + constant$, for all x, y. z_t would thus contain all the information we want (usually we are only interested in the variation of z_s, and do not care about the constant) (Fig. 1B, C).

For simplicity we model $s(z_t - z_s)$ with the configuration of an atomic force microscopy (AFM) (Fig. 1), in which

$$s(z_t - z_s) = \max\{0, -A((z_t - z_s) - z_0), A > 0\}.$$

Here z_0 is the difference in height between the tip and the sample when they just come into contact. The above formula is from the fact that, for small deflection, the change in laser beam position s is linearly proportional to $(z_t - z_s)$ (which is in turn proportional to the tip-sample contact force, since the cantilever is a linear spring for small deflection). The deflection is obviously 0 when the tip is not yet in contact with the sample. Because we are free to choose the reference point for z_t, we can set z_0 to be 0. Also for simplicity we can assume all the quantities are already properly non-dimensionalized, and set $A = 1$[4]. Therefore the simplified formula for s becomes

$$s(z_t - z_s) = \max\{0, -(z_t - z_s)\}.$$

B. PI feedback and automatic parameter tuning

We implement a PI feedback control with manually adjustable parameters to demonstrate its behavior first. In this model, the tip raster scans on the sample surface in discrete time steps (δt), with the following update rule for z_t, given parameters (P, I) and the set point s_0:

$$z_t^{(i)} := z_t^{(i-1)} + P(s - s_0) + I \sum_0^t (s - s_0)\delta t.$$

Note that while in many common PID controllers the feedback output *replaces* the previous value, in an SPM it is usually *added* to the previous value. To simulate the real-world behavior of PI feedback, a small white noise term is added to the output of the simulator, i.e. $s = s(z_t - z_s) + \eta\epsilon$, where ϵ is a random variable with uniform distribution between $\pm\frac{1}{2}$, and η controls the magnitude of the noise.

Now we make a few assumptions to simplify our simulation. First, we will only use 1D line scans instead of full 2D scans to evaluate the performance of PI feedback (i.e. tip only moves in one direction x). This is because 2D scans are simply arrays of individual 1D scans. Second, we assume that each line scan takes 1 second, and our PI controller operates at 250 Hz. Therefore $\delta t = 4$ millisecond and the number of pixels in one scan is 250 (reasonable values for real SPMs). Third, we fix s_0 to be 1 and the noise magnitude η to be 0.2. This roughly corresponds to setting the tip-sample contact force at $\sim$1 nN (common practice in real experiments) with a $\pm$0.1 nN readout error (typical value, limited by the instrument).

Results with a randomly generated 1D sample topography and arbitrary combinations of PI parameters can now be generated by executing the PI feedback within the simulator. These results are used as the data set to evaluate the performance of the feedback. Brutal-force exploration of the parameter space is carried out to identify whether a globally optimal parameter combination exists and whether there are multiple local minima.

We then attempt to find the optimal PI parameters by fitting a small number of data points with a neural network with one hidden layer and 10/5 hidden sigmoid neurons, first in the full parameter space, and then within the perimeter of the global minimum of the first step. Back-propagation with Bayesian regularization and 70-30 cross-validation is used for training. Finding the minimum of the fitted smooth function is trivial.

C. MDP-based feedback

Finally we build a reinforcement learning model based on MDP for pixel-by-pixel control of tip-sample interaction without any explicit PI feedback. A discretized state ($s - s_0$, the value of error) and action (Δz_t, how much and what direction to move the tip for the next point) space is used. Initially the transition probabilities are assumed to be equal, and the value function V and policy Π random. Each line scan on a randomly generated 1D sample topography is a trial, after which the MDP is updated. A new optimized value function V^* is then obtained via value iteration, which implicitly defines the new policy Π^* which acts greedily on V^*.

III. RESULTS AND DISCUSSIONS

A. PI feedback with manually adjustable parameters

Results with four different combinations of (P, I) are shown in Fig. 2. Here $z_s(x)$ is a smoothed square wave with moderate height, representing the cross section of e.g. a common optical grating. Intuitively, if (P, I) is too large, the tip tends to overreact and starts to self-oscillate (this is the case even if we do not add the noise term); if (P, I) is too small, z_t cannot follow z_s; a moderate (P, I) results in z_t being a reasonably faithful representation of z_s. We are not going to discuss in detail the strategy of tuning (P, I) manually, because our goal here is to enable auto-tuning of (P, I), even without much *a priori* knowledge or experience.

B. Automatic tuning of PI parameters

One would naively think that it is a straightforward optimization problem with the sum of absolute error $J(P, I) = \sum_i |s^{(i)} - s_0|$ being the objective function (usually we can tolerate sharp but narrow spikes, but not moderate and broad peaks in feedback error, thus absolute value works better than squared), and there is no need to invoke any machine learning. However, the objective function cannot be written in closed form, and brutal force calculations with the simulator (Fig. 3) show that although a robust global minimum exists, the objective function is non-convex due to the existence of numerous local minima that depend on the detailed topography (Fig. 3b).

We propose to implement a regression with a moderate number of points $\{((P^{(j)}, I^{(j)}), J(P, I)^{(j)})\}$, each j from a randomly generated line topography (z_s), and then find the global minimum of the fitted function $J^*(P, I)$. One can iterate in an increasingly small parameter space if necessary. The motivation is two-fold: First, in a real experiment it is impractical to try hundreds of thousands combinations to map out the full parameter space as in Fig. 3b. It is however easy to do a 2D scan with a few hundred lines with different parameters for each line. Second, by fitting with a relatively small number of points from randomly generated lines, we may avoid the shallow, small-scale and detail-dependent local minima and obtain a function $J^*(P, I)$ that is more suitable for finding the global minimum (in a real sample there are always random topography variations between lines,

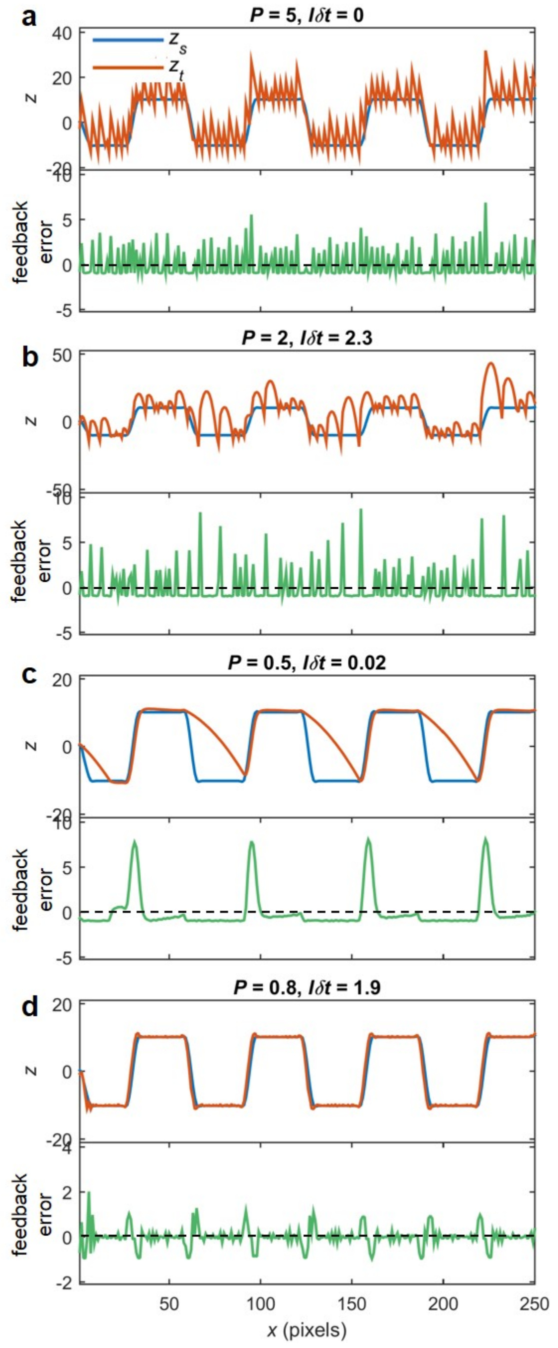

FIG. 2. Results of PI feedback simulation with different combinations of (P, I) values. **(a, b)** If P and/or I is too large, the tip will self-oscillate, with characteristics depending on the specific combination. **(c)** If P and I are too small, the tip cannot track the sample surface well. **(d)** An optimal combination of (P, I) makes z_t a faithful representation of z_s. z_t is offsetted for ease of comparison. Note that the feedback error is $s - s_0$ instead of $z_t - z_s$: The latter is unknown in real experiments.

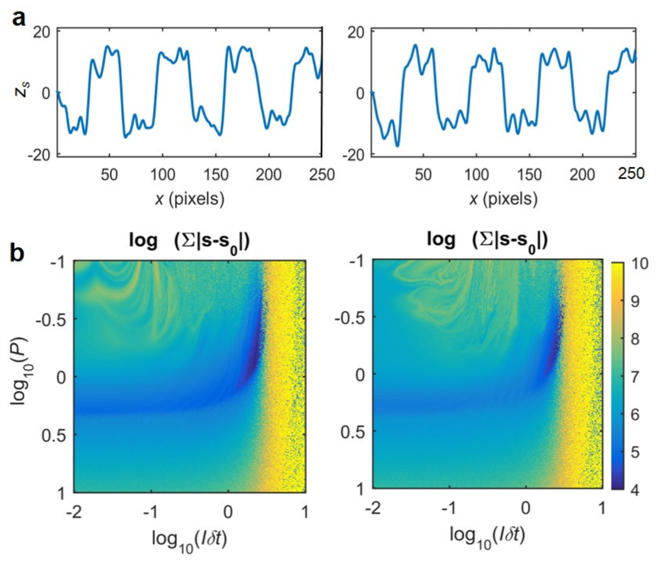

FIG. 3. The true topography (z_s) **(a)** and log-sum-error vs. (P, I) plot **(b)** for two randomly generated lines. The other parameters are identical to those used in Fig. 2. The plots in (b) are 400 by 400 spanning a large parameter space. Note that log-sum-error function is non-convex in small scales, with local minima depending on the details. Whereas the global minimum is robust and well defined in large scales.

which will serve the same purpose as the randomness introduced in this simulation).

We use a neural network with one hidden layer for the regression. Neural networks are universal function approximators and are particularly suited for fitting highly nonlinear functions. The specific procedure is as follows: First, 400 randomly generated line topography (z_s) and randomly picked (P, I) are fed into the simulator with PI feedback. The feature matrix is simply all the (P, I) values. The simulator returns the log-sum-error, which is recorded in the target matrix. Both matrices are then used to train a neural network with one hidden layer and 10 hidden neurons, using back-propagation, Bayesian regularization on the weights, and 70-30 cross-validation (Fig. 4a, b). The global minimum of the fitted function is (easily) found, and a second iteration finds the precise minimum with 200 additional data points and a second neural network with 5 hidden neurons (Fig. 4c, d). Overall the neural network fitting agrees well with the "true value" in Fig. 3b. (P, I) selected this way results in excellent performance (Fig. 4e).

We emphasize that such auto-tuning procedure could be highly practical and useful in real experiments. Scanning at 1 Hz, the 600 lines needed for a good result only takes 10 minutes (the time for training and minimum finding are negligible in comparison), a very reasonable time for a typical hour-long SPM session. In addition, the optimal (P, I) for different tip type, controller setting, scan speed and sample roughness can be very different, but by obtaining training examples under the exact same

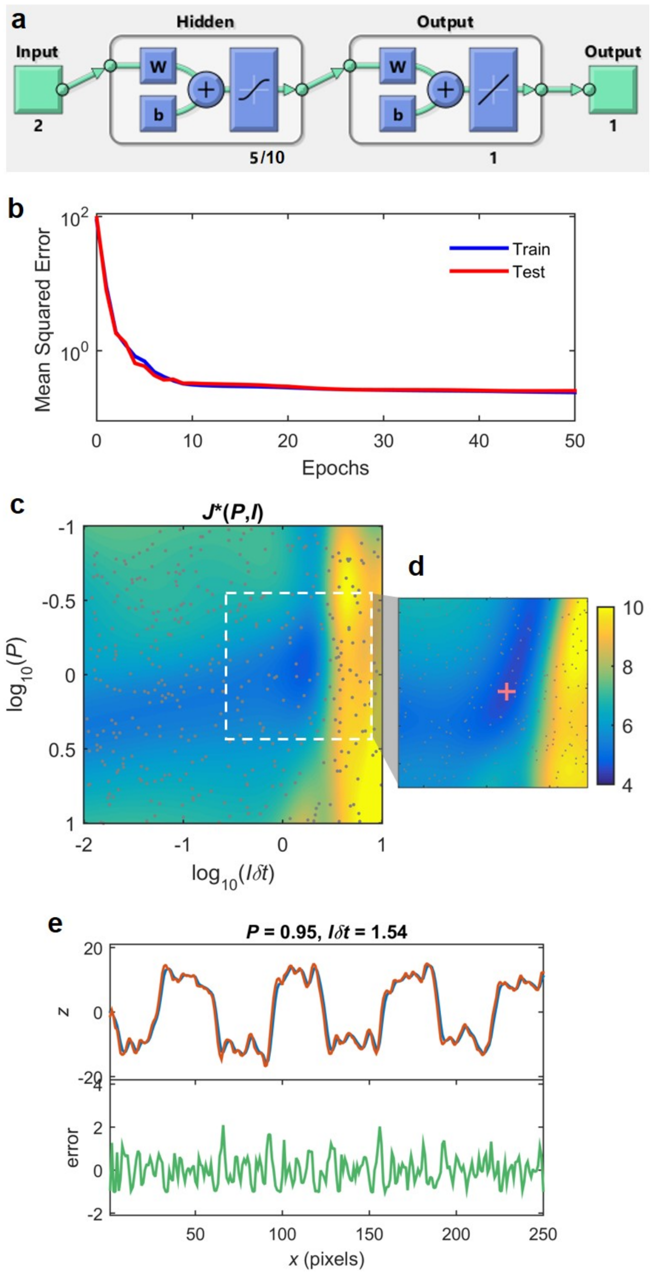

FIG. 4. Automatic PI parameter tuning based on neural network. **(a)** The neural network consists of one hidden layer with 5/10 hidden neurons. **(b)** Typical training curve with 400 data points and 10 hidden neurons. Back-propagation with Bayesian regularization and 70-30 cross-validation is used for training. **(c)** Regression result with 400 data points (grey) and 10 hidden neurons, showing good agreement with those in Fig. 3b. **(d)** Second iteration centered around the minimum of the first pass result, with 200 data points (grey) and 5 hidden neurons. The fine minimum is labeled by the red cross. **(e)** Scan performance on a randomly generated line topography with optimal (P, I) from (d). z_t tracks z_s very well.

condition as real data taking, the selected (P, I) is guaranteed optimal for the specific measurement.

We note that several other methods (e.g. biased random walk) may also overcome the many local minima to arrive at the global minimum, likely at a faster speed. The sampling+fitting method may be more robust overall, but a real-world implementation is needed to evaluate and make conclusive comparisons.

C. MDP-based reinforcement learning feedback

In this section we explore the possibility of using reinforcement learning to realize pixel-by-pixel feedback control without any explicit feedback model (e.g. PID feedback). At a high level, it is similar to the invert pendulum problem. The physical dynamics is in fact simpler, as no derivatives of location coordinates are involved. However the random variation of z_s between pixels (time steps) may pose a significant challenge to efficient learning. Ideally the model should learn about both the tip-sample interaction and the feature of z_s, to achieve good tracking.

The MDP model is setup as in Table I.

TABLE I. MDP setup.

S	A	P_{sa}	γ	R
error $(s - s_0)$ discretized	Δz_t discretized	Depends on tip-sample interaction and z_s	0.5-0.9	$-\lvert s - s_0\rvert$

The implementation of the MDP follows the description in Methods. A decision (how to update z_t) is made according to current state (error value) and policy at each pixel; the MDP and the associated optimal value function and policy are updated after each complete line scan. The typical results with the states being $s - s_0 \in [-2, 5]$ discretized into 30 states, and actions being $\Delta z_t \in [-5, 5]$ discretized into 30 states, and $\gamma = 0.9$ are shown in Fig. 5.

It is evident that in general the model does track z_s, and the error does get smaller after more trials. But the learning is quite stochastic and seems to stop after a moderate number of trials (Fig. 5(a)), although an optimal performance has not been achieved (compare the bottom panel in Fig. 5(d) with that in Fig. 4(e)). Having more states in state or action space, tuning γ or going to a continuous-value state space does not qualitatively solve the issue, and more often makes the learning even less robust.

One of the major causes of the non-ideal performance is likely the randomness in z_s. Going back to the inverted pendulum problem, it is obvious that if at every time step, a random offset of pendulum angle or cart position comparable to the actual increment due to physics is added, the MDP will not be able to learn or perform well.

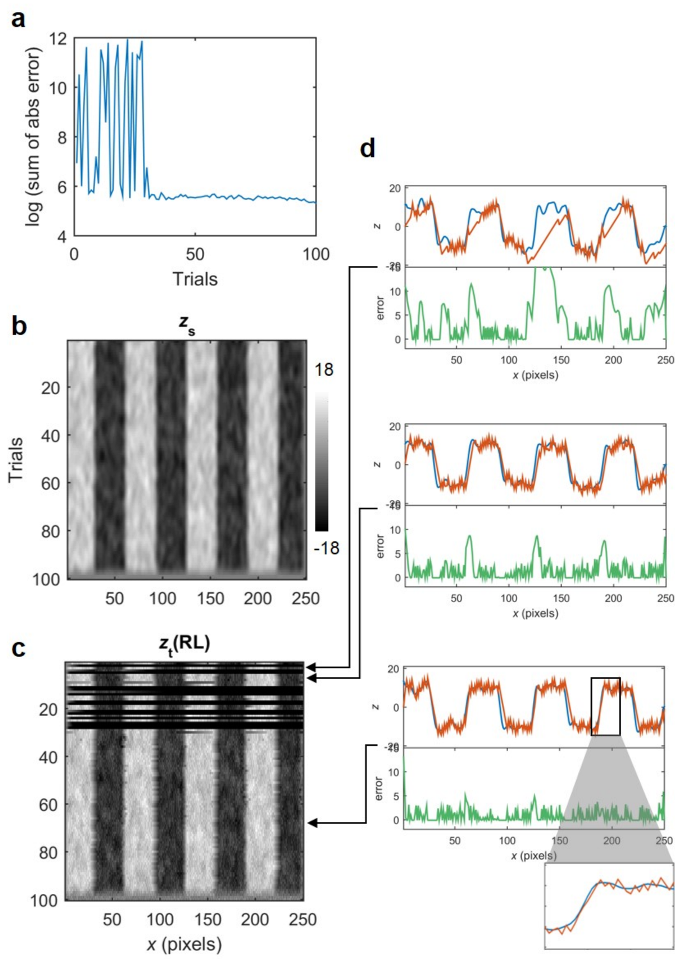

FIG. 5. Typical results with MDP-based feedback. **(a)** A typical learning curve. **(b)** A randomly generated 2D z_s map, as the "true value" topography for the MDP model to track. It contains 100 lines, hence 100 trials. **(c)** The resulting z_t. **(d)** The line plots at various stages of the learning process, as indicated by the arrows in (c). Inset shows the pixel-by-pixel behavior of z_t.

Similarly in the SPM case, the error due to the unpredictable z_s at the current point is comparable to that due to tip-sample interaction and the action taken in the last point, therefore instead of learning about *both* the tip-sample interaction and the feature of z_s, the model ends up learning about *neither*. A possible way to improve is to learn at a single point first, but doing so would be equivalent to measuring the tip-sample interaction and feed forward in subsequent scans. The potential flexibility to accommodate different scan speed and sample feature is also completely lost. Therefore the lesson learned is that an MDP-based reinforcement learning model cannot learn well in a system with excess noise due to the intrinsic randomness in the transition probabilities (e.g. stock market). Certain form of partially observed MDP (POMDP) may mitigate the problem, but is beyond the scope of this work.

IV. CONCLUSION AND FUTURE WORK

PI feedback is still the predictable, robust feedback method for SPM. A simple neural network fitting of a few hundred line scans implemented in this work can reliably find the optimal (P, I) parameters in a large parameter space, making it practical for a wide range of tip/sample conditions.

A simple MDP-based model on the other hand, does not offer the same level of robustness and performance. More advanced reinforcement learning model might be needed.

[1]K. H. Ang, G. Chong, and Y. Li, "Pid control system analysis, design, and technology," Control Systems Technology, IEEE Transactions on **13**, 559–576 (2005).

[2]X. Zhou, X. Dong, Y. Zhang, and Y. Fang, "Automatic tuning of pi controller for atomic force microscope based on relay with hysteresis," in *Control Applications,(CCA) & Intelligent Control,(ISIC), 2009 IEEE* (IEEE, 2009) pp. 1271–1275.

[3]D. Y. Abramovitch, S. Hoen, and R. Workman, "Semi-automatic tuning of pid gains for atomic force microscopes," Asian Journal of Control **11**, 188–195 (2009).

[4]For the more technical-minded, this is equivalent to making all the location values in nm (nanometer), s in nN (nano-newton) and the spring constant of the cantilever 1 N/m.