2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Assessing Internal Models for Faster Learning of Robotic Assembly

Jeremy A. Marvel, *Student Member, IEEE*, and Wyatt S. Newman, *Senior Member, IEEE*

Electrical Engineering and Computer Science Department
Case Western Reserve University
Cleveland, OH 44106, USA

{jeremy.marvel & wyatt.newman}@case.edu

***Abstract* – This work investigates what makes a robotic assembly process "learnable" for the explicit purpose of improving the performance of that process. It has been observed that even stochastic search methods like Genetic Algorithms (GA) can benefit from advanced models of the assembly task. Models built from the results of random samplings of a parameter space have been used previously to predict the performances of parameter sequences not yet evaluated, but the question of what properties of the models actually benefit the optimization remained. A quantitative analysis algorithm is derived and tested on physical assemblies for validation. Results are provided that illustrate the efficacy of the analysis algorithm for prediction-based performance enhancement when such models are used.**

***Index Terms* – Model building, parameter optimization, robotic assembly**

I Introduction

THIS paper concerns algorithms that enable a robot to improve its performance in mechanical-assembly tasks autonomously through exploration. It has been shown in previous work on several automotive powertrain assembly tasks that a compliantly-controlled robot can tune its own assembly programs to achieve high levels of skill [1]. Subsequent to this work, it was proposed that the construction of an internal model predicting task performance as a function of tunable parameters would enable faster learning. In experiments, instances of success in this approach were found. However, in some other cases, the internal model did not seem to express marked benefit. In the present work, we help to clarify why and when internal models can speed up autonomous learning of mechanical assembly.

One expectation is that a high-quality model can benefit a robotic process by providing a reliable medium for offline optimization and simulation. A low-quality model, however, has the potential to actually be worse than no model at all, because it may present to the system inaccurate information or even information that is contrary to the truth. Being able to distinguish between the two, however, becomes problematic without a third, more knowledgeable model for comparison. Thus the problem manifests: what is the quality of the predictive model, and by what metric does one compare two different models?

Clearly, the latter question may be addressed by running various parameter sequences through each model being compared and then evaluating the same parameters in the physical system. Whichever model performs closer to the physical results must naturally be the better of the two. However, such an approach is naïve, and does little to quantitatively describe either the quality of how well either model actually captures the parameter-performance mapping or whether or not the system is even capable of being learned.

Recent research [2] has shown that simple numerical models could effectively improve stochastic searches for robotic assembly parameter optimization by discarding those parameter sequences computationally deemed unlikely to result in improved assembly performances. There was little *a priori* indication, however, which of the simple models used would be of more assistance in improving the performance of the stochastic search, or even whether or not either would be of *any* assistance. To that end, in this study we introduce a metric for determining the quality of a predictive virtual model. Models created dynamically for predicting the performance of a robot running peg-in-hole assembly searches over a variety of parameter spaces are compared according to this metric, and then evaluated for validation.

II Development of Internal Models

Mechanical assemblies can be parametrically described as sequences of search strategies. These searches can be autonomously optimized for the minimization of time and contact force by being evaluated by a GA [3]. The search process, however, is wasteful since knowledge of evaluated sequences is lost upon the subsequent generation of testing. By building up a model that approximates the mappings of input parameters to their respective output performances, this knowledge can be preserved and used to guide the GA evolutionary process by evaluating potential child gene sequences and selectively pruning the population of any sequences deemed unlikely to produce good assembly results..

There are a number of approaches for creating virtual models of physical assemblies. Abstract models of task spaces have been created that describe assemblies in terms of the total entropies of the system [4], but focused primarily on parts acquisition, orientation and positioning rather than the actual process of putting components together. Also developed are models based around the concept of "assembly features" to design and describe assembly sequences [5], manufacturing processes [6], and assembly "intent" [7]. These approaches, however, are focused on operator-driven assembly planning and design, and, again, do not address the process of joining pieces.

Other methods of modeling are not so abstract. Simulators created prior to evaluative training for process learning [8] and parameter optimization [9] have been successfully employed for offline learning. The inline generation of models has also been utilized to extrapolate information in order to dynamically generate 3D representations of the assembly components [10] while others extract more abstract physical characteristics of complex sensor reactions in effect within the system [11]. In our previous work, models took the form of dynamically-produced mappings that linked the assembly parameters with the resulting performances. In the context of the remainder of this paper these are the models being analyzed.

For simplicity, we are utilizing standard feed-forward artificial neural networks (ANNs) with back propagation to generate our model of the system. Training data consists of randomly-generated input parameters and their expected (averaged) resulting assembly performances when evaluated *K* times by a robot system. In the grander scheme of this research, however, any model that attempts to explain the assembly problem for the purpose of predicting robot performance would be suitable. The model, as it is applied to a GA, is utilized as a predictive filter in the sense that it selectively prunes the child gene population to a fractional subset consisting of the top sequences that it projects will perform better than the rest. Ideally, with a proper model, a stochastic method could effectively have the same performance as a standard gradient descent when provided with competent guidance. The actual formulations of the ANN and GA implementation are beyond the scope of this paper and are thus not covered here. Their implementations, however, are discussed in detail in [2].

Having a perfect model eliminates the need for parameter evaluation, as everything can be computed *in silico*. However, assuming one has only imperfect knowledge of the system necessitates the evaluation of the parameter sequences in order to test and compare the performance of the system. In terms of robotic assembly no model will ever be perfect due to the noise inherent in the system. Noise takes the form of position and orientation uncertainty, minute variations from one part to another, tool and robot wear, and friction. Thus a model must be able to overcome this noise in order to be of substantive use.

III Formulation of the Quality Hypothesis

With numerous possibilities for model implementation and design, the most effective aspect available for comparison is the set of outputs produced by the model for a set of input parameters. When charted, the inputs create a multidimensional surface plot for each output. It is hypothesized that the output surfaces of a good quality model will possess the following traits: 1) they capture the empirical evidence accurately, 2) they are not horizontal planes, and 3) they express low spatial frequency. Certainly additional properties exist that can be compared, but currently only these three aspects are investigated.

The first trait is, of course, absolutely mandatory because any model that cannot explain what has already been seen is of little use as a future performance predictor. Here we define the quality of the fit to the data by the RMS error, *R*, between the predicted output, *o*, and the actual average performance output, *t*, as computed by Equation 1 for each of the *M* previously evaluated samples *i*.

$$R = \sqrt{\frac{\sum_i^M (t_i - o_i)^2}{M}} \qquad 1$$

The RMS value computed in Equation 1 can be thought of as the model fit standard deviation. In contrast we can define the standard deviation—or scatter—of the repeated trial results in terms of empirical data by Equation 2. Here the values σ_i are computed as the standard deviations for the *K* trials performed for each of the *M* input parameter sequences.

$$b = \frac{\sum_i^M \sigma_i}{M} \qquad 2$$

In the context of this study, we wish to utilize the metric of model comparison as a scoring function, with the performance of the "better" model thus having a higher score than the "worse" model. Further, we wish to force the fitness metric to conform to the form of being a unit-less scalar value in the range [0, 1], with 1 being perfect fitness. As such, we define the surface fitness, *d*, as being the sigmoid computed based on the ratio of the model fit and trial scatter standard deviations by Equation 3.

$$d = 1 - e^{-b/R} \qquad 3$$

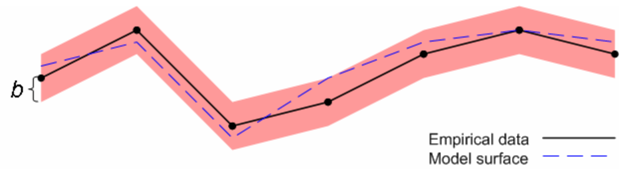

Fig. 1. Illustration of the model fit to the 1D empirical data with reference to the natural variance, *b*.

The empirical data, when plotted out, can be thought of as a surface with thickness *b* as shown in Fig. 1. The quality of fit to the data can be equated to how well the model adheres to the center of this surface.

The second requirement—that the model is not a horizontal plane—needs some clarification because, in high dimensions, "horizontal" is an ambiguous and largely unhelpful term. Here, we define horizontal as being the property that there exist no adjustments of values in the parameter space that will result in a discernable performance difference in the evaluative process. The quality that the model is not a horizontal plane stems from the desire to be able to use the model as a predictor for assembly performance improvement. If adjusting parameters do not result in changes in performance, there is little benefit by addressing the model. Doing so would likely exhibit a performance on par with an unassisted random search, which would mean the resources expended to develop and reference the model were essentially wasted.

This requirement does not preclude the possibility of flat planes in general, as transitions in any direction that result in improved performances are demonstrably optimizable through simple hill-climbing algorithms. The range of outputs of the model is irrelevant for this value, however, as what is being tested is the binary state of whether or not the output surface plot is a horizontal plane (Fig. 2).

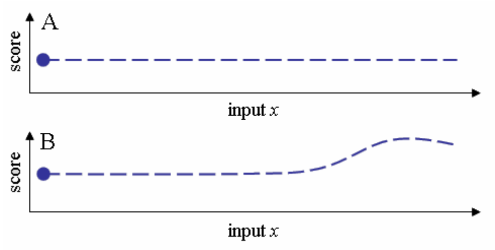

Fig. 2. 1D model exhibiting horizontal plane behavior (A) versus a model demonstrating some benefit to adjusting parameters (B).

We can describe this as being the property that the potential improvement difference between the observed maximum and minimum projected outputs of the model ($o_{\max}$ and $o_{\min}$ respectively) must be significant in relation to the natural variations exhibited between physical trials of identical parameter sequences, as seen in Equation 4. If the projected performance improvement between the modeled worst and best possible parameter sequences is dwarfed by the expected level of noise experienced simply by evaluating the parameter sequences, there is arguably little to be gained by modeling the system.

$$u = o_{\max} - o_{\min} \qquad 4$$

Once again, we wish to determine the benefit to be gained relative to the natural variance between trials given identical parameter sequences and assess this value in the range [0, 1]. This horizontal metric is thus computed by Equation 5.

$$p = 1 - e^{-u/b} \qquad 5$$

The third constraint, low spatial frequency, originates from the observation that gradient descent performs best when the surface function produces a smooth transition from a given coordinate to neighboring points as evaluated over large areas. Surface plots with high spatial frequency provide little information in the way of nearsighted performance trends, and as such, parameter optimization is largely left to trial-and-error and random chance. Gradient trends over large search spaces may never be discovered due to local optima that distract short-sighted search algorithms. And, although high-dimensional mesas (that is, a range of "sweet spot" parameters for which highly optimal performances are guaranteed but around which successful assembly may be impossible) are feasibly learnable, they frequently cannot be discovered by trend searches.

The smoothness error of the N-dimensional model surface, defined in Equation 6, computes the summed output error based on a multi-dimensional low-pass surface filter mask and the actual projected outputs across the model surface. In short, it computes the running average point difference for all model outputs $o_{i_1, i_2, \ldots, i_N}$. This value is distinct from the RMS error computed in Equation 1 in that only the model surface is investigated while the actual trial results are ignored. Worth noting is that while the number of positional samples is accounted for in the equation by the value of a, the step size between neighboring points along the multi-dimensional surface is ultimately user-defined.

$$E = \sqrt{\frac{\displaystyle\sum_{i_1=0}^{I_1} \sum_{i_2=0}^{I_2} \cdots \sum_{i_N=0}^{I_N} \left(\frac{\displaystyle\sum_{j_1=i_1-a}^{i_1+a} \sum_{j_2=i_2-a}^{i_2+a} \cdots \sum_{j_N=i_N-a}^{i_N+a} o_{j_1, j_2, \ldots, j_N}}{(2a+1)^N} - o_{i_1, i_2, \ldots, i_N} \right)^2}{\displaystyle\prod_{i=0}^{N} I_i}} \qquad 6$$

In the 1-dimensional case, for example, the equation for E would look like the following:

$$E = \sqrt{\frac{\displaystyle\sum_{i=0}^{I} \left(\frac{\displaystyle\sum_{j=i-a}^{i+a} o_j}{2a+1} - o_i \right)^2}{I}}$$

Continuing with the 1-D example, shown in Fig. 3 are two simulated surface models generated for spatial frequency testing. The model surface of the top plot shows a higher spatial frequency relative to that of the bottom plot. Because of this, given a = 1, the low-pass filtered surface model does not fit the data as well as it does the lower plot. This is reflected in the smoothness errors E_A = 0.812 and E_B = 0.238.

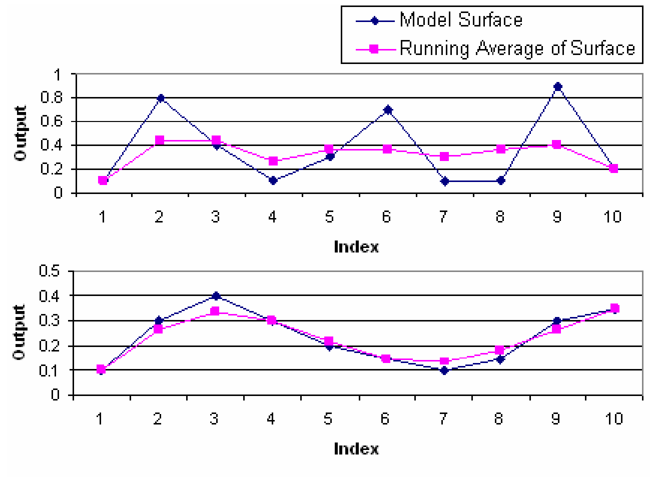

Fig. 3. Sample 1D plots showing the difference between the actual and running average surfaces for high (top) and low (bottom) spatial frequency models.

The frequency score value, f, is thus computed based on the ratio between the natural variance and the smoothness error as defined in Equation 7.

$$f = 1 - e^{-b/E} \qquad 7$$

The three qualities of the model output surface are all effectively interrelated. A model cannot express high spatial frequency on its surface and be a horizontal plane, for example. Similarly, if the model accurately captures all of the empirical data and is simultaneously a horizontal plane, it is apparent that no level of modeling will benefit the optimization process, as one would likely deduce that there is nothing that would result in a performance any better or worse than any other. We define the quality metric, $q \in [0.0, 1.0]$, as the product of the capture score, non-horizontal planar requirement, and spatial frequency score according to Equation 8.

$$q = dpf \qquad 8$$

The nonlinear nature of q is necessary for this model quality assessment metric. Though a linear sums method would also peak as the three values of d, p and f approach 1.0, it would not permit any one term to have veto power (that is, to declare the model as being of low quality) over the other two. For this reason, we require the numerical equivalent of the logical AND. The value of q gives us not only a metric for qualifying a given model, but also provides a comparative means for assessing which of a number of models is more likely to produce better performance predictions. For any two given models m_1 and m_2, $m_1 \neq m_2$, the model m_1 is considered to be the better of the two if, based on their respective quality metrics q_1 and q_2 for the same output unit (ex. assembly speed, applied force, etc.), the value $q_1 > q_2$.

IV Experimental Results

In order to validate the quality hypothesis, a physical trial configuration has been set up to test the projected benefits of utilizing dynamically-generated internal models for assembly. Initial tests performed modeling for the purpose of assisting the assembly parameter optimization for a robot joining components of a transmission valve body from an automotive automatic engine.

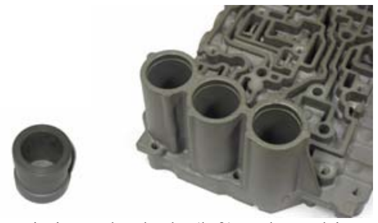

Fig. 4. Transmission valve body (left) and spool insert (right) to be assembled by the robotic manipulator.

The assembly largely consists of a single spiral search that performs a peg-in-hole insertion of a metal spool into a high-tolerance opening, as seen in Fig. 4. When initiated, the spiral search (Fig. 5) behavior begins moving the tool from its current location. Applying a constant downward force, the tool center point gradually moves outward along the XY lateral plane in a spiral pattern until the commanded spiral has been completed. Once this occurs, the spiral search reinitiates from its current location. This search behavior repeats until either a restart or termination condition has been met. The search terminates either upon a timeout or on successfully meeting the needs for insertion completion. In these trials, assembly success is determined by effectively reaching a specified insertion depth. Upon termination, the motion of the robot halts and the controller triggers a request for the next gene in the sequence.

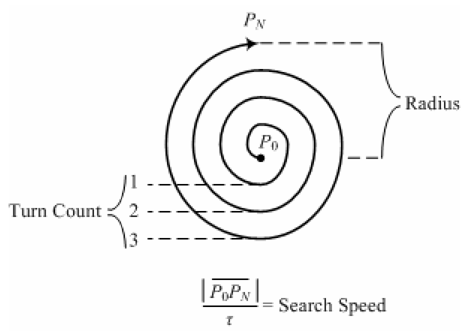

Fig. 5. Spiral search parameter description.

Parameters from the spiral search that can be optimized include the downward applied force, the rotational search speed, spiral search radius, and the number of turns per spiral. For this early research, we are concentrating only on the spiral speed, radius and number of turns, and are maintaining the downward force constant at 5 N. In order to provide a visualization of the multidimensional input space, the search parameters were separated into three set pairs and plotted individually such that their relationship with one another can be illustrated. The remaining, unevaluated parameter for each model had its value locked and was not modified. Each of the three model environments (speed-vs-radius, speed-vs-turns, and radius-vs-turns) were passed through the same model-building process (i.e. training an ANN on assembly trial data), the results of which are evaluated for predicted performance.

Because the assembly was fixtured in place, prior to each trial nontrivial noise in the form of position uncertainty was simulated by adding a random, lateral offset in the range of ±1mm on both the X and Y axes. Each parameter sequence was evaluated numerous times, with each evaluation beginning with a new random offset. The resulting performances of the iterations of each given parameter sequence were then averaged and associated with the parameter sequence.

Stochastic searches were performed using two different method: unassisted searches, and model-enhanced assisted searches. The unassisted stochastic search was configured as a GA exploration of the parameter space that used random perturbations of the parameter "gene" sequences to drive optimization. Upon each successive generation, 10 random child gene sequences were produced based on the principle of mutating the parent gene. The best-performing child, as determined by empirical evaluation, was then selected to be the subsequent generation's parent. The assisted search was identical to the unassisted method, with the exception that 1,000 random child gene sequences were produced on each generation, and that the top 10 child genes projected to perform the best were then selected for evaluation and possible parental succession.

Gene sequences were expressed as vectors of floating point numbers that describe the search parameters and termination conditions for evaluation. These sequences are evaluated by being passed through an application that interpreted the genes and issued commands to an ABB IRB-140, 6-DOF open-chain manipulator outfitted with an ATI Gamma force/torque sensor.

Initial data for the ANN was generated from 200 random parameter samples for each model that were evaluated through the robot to produce empirical results. In each model case, the ANN topology consisted of 20 input layer nodes (the length of a single search descriptor gene sequences from [2]), 10 hidden layer nodes, and a single output layer node. With low dimensionality of hidden-layer neurons, the surface plots would be inherently smooth. Thus the *d* and *p* terms would be the most influential when determining the values of *q*. In higher dimensions, however, the *f* term becomes more dominant. The utilization of 10 hidden-layer neurons allows for the possibility of higher spatial frequencies while still effectively forcing an abstraction of the training data.

The ANNs were then trained for 5,000 epochs. The surface plots generated by the networks are illustrated in Figures 5-7. In each plot, the inputs and outputs have been normalized to be in the range [-1.0, 1.0]. Here, the "score" value is a function of the assembly time with respect to a maximum allowed value. If an assembly attempt exceeds this value, the robot considers the attempt a failure and aborts the search process.

For these initial trials the value of $o_{\min\Delta}$ for the computation of the *p* term in Equation 3 was set to 0. For the computation of the spatial frequency equation, *f*, the value of the *a* term was set to 1, and surface samples were taken at 0.1 unit intervals along each axis. The computed surface properties are presented in Table 1, and the resulting model quality metrics are given in Table 2.

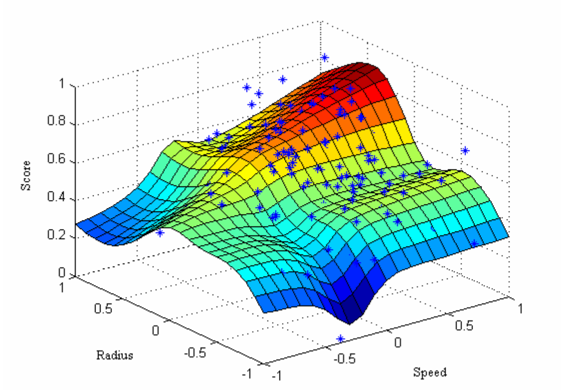

Fig. 5. Surface plot produced by the ANN based on the parameter samples randomly varying the spiral speed and radius search values.

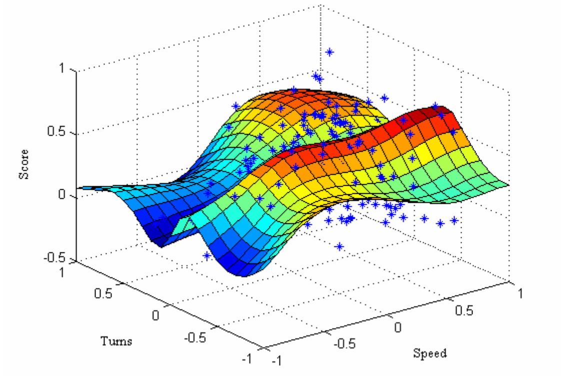

Fig. 6. Surface produced by randomly varying the spiral speed and number of turns search values.

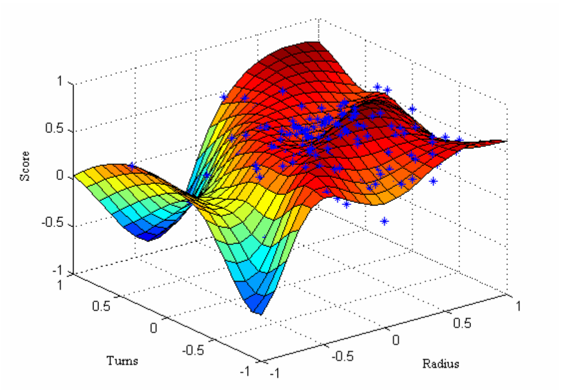

Fig. 7. Surface plot produced by randomly varying the spiral radius and number of turns search values.

	***b* (s)**	***u* (s)**	***R* (s)**	***E* (s)**
Speed vs Radius	3.547	6.068	1.125	0.315
Speed vs Turns	4.198	6.197	2.165	2.040
Radius vs Turns	4.181	5.836	1.355	2.760

Table 1. The model surface properties with regard to assembly time for the three sample network surface topologies.

	D	***p***	***f***	***q***
Speed vs Radius	0.957	0.819	1.000	0.784
Speed vs Turns	0.856	0.771	0.872	0.575
Radius vs Turns	0.954	0.752	0.780	0.560

Table 2. The model quality properties of the three sample network surface topologies.

Based on the results of applying our quality metric to the three models, one would suspect that the model performance of the search speed versus the search radius would out-perform the other two models. One might further expect that the benefit of the speed-versus-turns and radius-versus-turns models would be negligible in terms of improvement over their respective unassisted parameter searches given their low scores for *q*.

To test these results, both the assisted and unassisted GA implementations were evaluated five times each for eight generations of training. After each generation (10 trials per generation), the best-performing child's assembly time was selected to be indicative of the implementation as a whole for that generation. The assembly times for the best-performing child at each generation marker were then averaged, and plotted in Figures 8-10. The error bars in the graphs represent a single standard deviation in performance above and below the average.

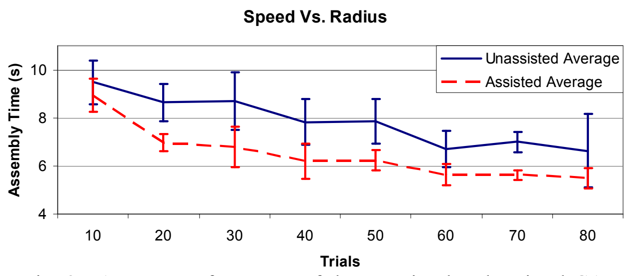

Fig. 8. Average performances of the unassisted and assisted GA implementations for optimizing the search speed and spiral radius parameters.

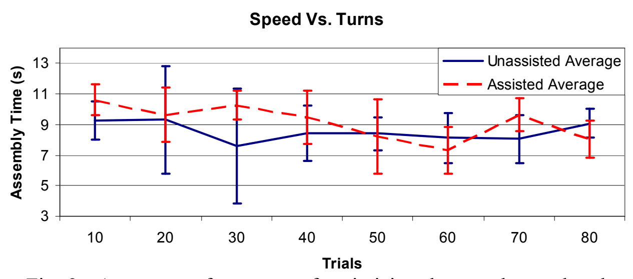

Fig. 9. Average performances of optimizing the search speed and number of turns parameters.

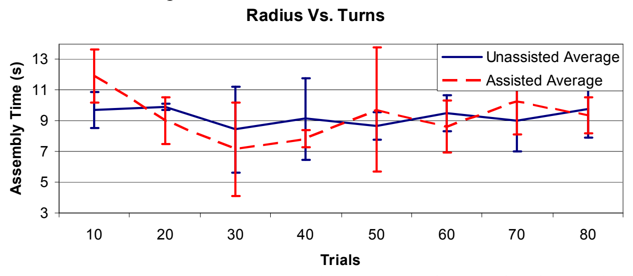

Fig 10. Average performances of optimizing the spiral radius and number of turns parameters.

As predicted by the performance metric, use of the speed-vs-radius model demonstrates improvement over the unassisted stochastic search. In contrast, the speed-vs-turns and radius-vs-turns exhibited highly variable performance without any real indication that their applications to stochastic searches perform any better than their respective unassisted Genetic Algorithms implementations.

V Discussion

Knowing in advance whether or not a model can be of benefit to an optimization process, or even whether that model is capable of capturing and explaining the performance of the system, is valuable. This paper outlined an approach for quantitatively evaluating a model based on its multi-dimensional surface plots. The ultimate goal of this research was to develop a comprehensive strategy to assess both the efficacy and benefit potential of virtual models of robotic assemblies.

Additional development of this assessment metric is likely to aid process engineers in the determination of which model strategies to adopt for different assembly problems, whether or not the application of an internal model is even warranted, and potentially the extent and form of the allocation of resources to modeling and optimization. Future work will attempt to automate the model analysis process, effectively allowing for dynamic mode switching and progress recognition.

An interesting additional observation to be made is that the unassisted Genetic Algorithms implementations for the speed-vs-turns and radius-vs-turns model tests exhibited little improvement over time. It is envisioned that, with additional tuning, this metric will provide a means of determining whether a process is learnably optimizable—that is, if an intelligent algorithm can detect and take advantage of trends in the mapping of parameters to performance—or if optimization must be completed by trial-and-error approach, or even if the process can be optimized at all. One proposed embodiment would determine a trust metric for the predictive models that could be adjusted before the models are even utilized.

Acknowledgments

Support for this research was provided by the Robot and Automation division of the ABB Inc. U.S. Corporate Research Center.

References

[1] Jing Wei and W. S. Newman. "Improving Robotic Assembly Performance Through Autonomous Exploration." Proceedings of the 2002 IEEE International Conference on Robotics and Automation. 2002. Vol. 3. Pp. 3303-3308.

[2] J. Marvel and W. Newman. "Accelerating Robotic Assembly Parameter Optimization Through the Generation of Internal Models." Proceedings of the 2009 IEEE Conference on Technologies for Practical Robotic Applications. 9-10 November, 2009.

[3] J. Marvel, et al. "Automated Learning for Parameter Optimization of Robotic Assembly Tasks Utilizing Genetic Algorithms." Proceedings of the 2008 IEEE International Conference on Robotics and Biomimetics. 21-26 February, 2009. Pp. 179-184.

[4] A. Sanderson. "Parts Entropy Methods for Robotic Assembly System Design." Proceedings of the 1984 IEEE International Conference on Robotics and Automation. March, 1984. Pp. 600-608.

[5] D. Deneux. "Introduction to Assembly Features: An Illustration Synthesis Methodology." Journal of Intelligent Manufacturing. Vol.. 10. 1999. Pp. 29-39.

[6] A. Gayretli and H. Abdalla. "A Feature-Based Prototype System for the Evaluation and Optimization of Manufacturing Processes." Proceedings of the 24th International Conference on Computer and Industrial Engineering. Vol 37. October, 1999. Pp. 481-484.

[7] H. Ullah, E. Bohez, and M. Irfan. "Assembly Features: Definition, Classification, and Instantiation." IEEE 2006 International Conference on Emerging Technologies. 13-14 November, 2006. Pp. 617-623.

[8] A. Ng, *et al.* "Autonomous Helicopter Flight via Reinforcement Learning." Advances in Neural Information Processing Systems. No. 16. 2004. Pp. 799-806.

[9] N. Yamanobe, *et al.* "Optimization of Damping Control Parameters for Cycle Time Reduction in Clutch Assembly." Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. 2-6 August, 2005. Pp. 3251-3256.

[10] S. Chhatpar and M. Branicky. "Localization for Robotic Assemblies with Position Uncertainty." Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems. October, 2003. Pp. 2534-2540.

[11] J. Deiterding and D. Henrich. "Automatic Adaptation of Sensor-Based Robots." Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. 29 October-2 November, 2007. Pp. 1828-1833.