Proceedings of the 1998 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Victoria, B.C., Canada • October 1998

Optimization of Human Control Strategy with Simultaneously Perturbed Stochastic Approximation

Jingyan Song[1], Yangsheng Xu[2,3], Yeung Yam[2], Michael C. Nechyba[3]

[1]Department of Systems Engineering & Engineering Management
[2]Department of Mechanical and Automation Engineering
The Chinese University of Hong Kong, Shatin, NT, Hong Kong
[3]The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA

Abstract

Modeling dynamic human control strategy (HCS) is becoming an increasingly popular paradigm in a number of different research areas, ranging from robotics to intelligent vehicle highway systems. Usually, HCS models are derived empirically, rather than analytically, from real human input-output data. While these empirical models offer an effective means of transferring intelligent behaviors from humans to robots and other machines, the models are not explicitly optimized with respect to potentially important performance criteria. In this paper, we therefore propose an iterative algorithm for optimizing an initially stable HCS model with respect to an independent, user-specified performance criterion. We first collect driving data from different individuals through a real-time graphic driving simulator. Next, we describe how we model each individual's control strategy through flexible cascade neural networks. Once we have initially stable HCS models, we propose simultaneously perturbed stochastic approximation (SPSA) to optimize these models with respect to a chosen performance criterion, Finally, we describe and discuss some experimental results with the proposed algorithm.

1 Introduction

HCS models, which accurately emulate dynamic human behavior, find application in a number of research areas ranging from robotics to the intelligent vehicle highway system. Because human control strategy (HCS) is a dynamic , nonlinear stochastic process, developing good analytic models of human control strategies tends to be difficult. Therefore, recent work in modeling HCS has focused on learning empirical models, through, for example, fuzzy logic [1, 2], and neural network techniques [3]. Since these HCS models are empirical, few if any guarantees exist about their theoretical performance. Thus, performance evaluation is an integral aspect of HCS modeling research, without which it is impossible to rank or prefer one HCS controller over another.

Performance evaluation, is, however, only part of the solution for effectively applying models of human control strategy. While humans are in general very capable of demonstrating intelligent behaviors, they are far less capable of demonstrating those behaviors without occasional errors and random (noise) deviations from some nominal trajectory. Any empirical learning algorithm will necessarily incorporate those problems in the learned model, and will consequently be less than optimal. Furthermore, control requirements may differ between humans and robots, where stringent power or force requirements often have to be met. A given individual's performance level, therefore, may or may not be sufficient for a particular application.

Hence, in this paper we propose an iterative optimization algorithm, based on simultaneous perturbed stochastic approximation (SPSA), for improving the performance of learned HCS models. This algorithm leaves the learned model's structure in tact, but tunes the parameters of the HCS model in order to improve performance. It requires no analytic formulation of performance, only two experimental measurements of a user-defined performance criterion per iteration. The initial HCS model serves as a good starting point for the algorithm, since it already generates stable control commands.

In this paper, we first introduce the dynamic graphic driving simulator from which we collect human control data and with which we investigate the modeling and evaluation of human control strategies. We then show how we model individual's driving control strategies using the flexible cascade neural network learning architecture. Next, we describe two performance criteria specifically related to the task of driving. In the following section, we then propose the iterative optimization algorithm for improving performance in the HCS models. Finally, we describe and discuss some experimental results with the algorithm.

2 Experimental setup

For this work, we collect human driving data from a real-time graphic simulator, whose interface is shown in Figure 1. In the simulator, the human operator has independent control of the vehicle's steering as well as the brake and gas pedals. The simulated vehicle's dynamics are given by the following second-order nonlinear model:

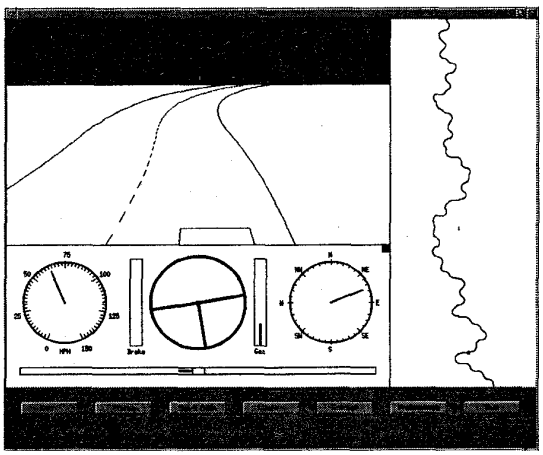

Figure 1: The driving simulator gives the user a perspective preview of the road ahead.

$$\ddot{\theta} = (l_f P_f \delta + l_f F_{\xi f} - l_r F_{\xi r})/I \quad (1)$$

$$\dot{\nu}_\xi = (P_f \delta + F_{\xi f} + F_{\xi r})/m - \nu_\eta \dot{\theta} - (sgn\nu_\xi) c_d \nu_\xi^2 \quad (2)$$

$$\dot{\nu}_\eta = (P_f + P_r - F_{\xi f}\delta)/m + \nu_\xi \dot{\theta} - (sgn\nu_\eta) c_d \nu_\eta^2 \quad (3)$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \nu_\xi \\ \nu_\eta \end{bmatrix} \quad (4)$$

where,

$$\dot{\theta} = angular\ velocity\ of\ the\ car, \quad (5)$$

$$\nu_\xi = lateral\ velocity\ of\ the\ car, \quad (6)$$

$$\nu_\eta = longitudinal\ velocity\ of\ the\ car, \quad (7)$$

$$F_{\xi k} = \mu F_{zk}(\tilde{\alpha}_k - (sgn\delta)\tilde{\alpha}_k^2/3 + \tilde{\alpha}_k^3/27) \times \sqrt{1 - P_k^2/(\mu F_{zk})^2 + P_k^2/c_k^2}, \quad k \in \{f, r\} \quad (8)$$

$$\tilde{\alpha}_k = c_k \alpha_k/(\mu F_{zk}), \quad k \in \{f, r\} \quad (9)$$

$$\alpha_f = front\ tire\ slip\ angle = \delta - (l_f\dot{\theta} + \nu_\xi)/\nu_\eta, \quad (10)$$

$$\alpha_r = rear\ tire\ slip\ angle\ = (l_r\dot{\theta} - \nu_\xi)/\nu_\eta, \quad (11)$$

$$F_{zf} = (mgl_r - (P_f + P_r)h)/(l_f + l_r),$$

$$F_{zr} = (mgl_f + (P_f + P_r)h)/(l_f + l_r), \quad (12)$$

$$\xi,\ \eta = body - relative\ lateral,\ longitudinal\ axis, \quad (13)$$

$$c_f, c_r = 50000N/rad,\ \ 64000N/rad \quad (14)$$

$$c_D = air\ resistance = 0.0005m^{-1} \quad (15)$$

$$\mu = coefficient\ of\ friction = 1,$$

$$F_{jk} = \ frictional\ forces, j \in \{\xi, \eta\},\ \ k \in \{f, r\} \quad (16)$$

$$P_r = \begin{cases} 0, & P_f \geq 0 \\ k_b P_f, & P_f < 0, \quad k_b = 0.34 \end{cases} \quad (17)$$

$$m = 1500kg,\ \ I = 2500kg - m^2,\ \ l_f = 1.25m,$$

$$l_r = 1.5m,\ \ h = 0.5m, \quad (18)$$

and the controls are given by,

$$-8000N \leq P_f \leq 4000N \quad (19)$$

$$-0.2rad \leq \delta \leq 0.2rad \quad (20)$$

where P_f is the longitudinal force on front tires, and δ is the steering angle.

We ask each individual to navigate across several randomly generated roads,which consist of a sequence of (1) straight-line segments, (2) left turns, and (3) right turns. The map in Figure 1, for example, illustrates one randomly generated 20km road for which human driving data was recorded. Each straight-line segment as well as the radius of curvature for each turn range in length between 100m and 200m. Nominally, the road is divided into two lanes, each of which has width $w = 5$m. The human operator's view of the road ahead is limited to 100m. Finally, the entire simulator is run at 50Hz.

3 HCS modeling

In this paper, we choose the flexible cascade neural network architecture with node-decoupled extended Kalman filtering (NDEKF) [6] for modeling the human driving data. We prefer this learning architecture over others for a number of reasons. First, no *a priori* model structure is assumed; the neural network automatically adds hidden units to an initially minimal network as the training requires. Second, hidden unit activation functions are not constrained to be a particular type. Rather, for each new hidden unit, the incremental learning algorithm can select that functional form which maximally reduces the residual error over the training data. Typical alternatives to the standard sigmoidal function are sine, cosine, and the Gaussian function. Finally, it has been shown that node-decoupled extended Kalman filtering, a quadratically convergent alternative to slower gradient descent training algorithms (such as backpropagation or quickprop) fits well within the cascade learning framework and converges to good local minima with less computation [6].

The flexible functional form which cascade learning allows is ideal for abstracting human control strategies, since we know very little about the underlying structure of each individual's internal controller. By making as few *a priori* assumptions as possible in modeling the human driving data, we improve the likelihood that the learning algorithm will converge to a good model of the human control data.

In order for the learning algorithm to properly model each individual's human control strategy, the model must be presented with those state and environmental variables upon which the human operator relies. Thus, the inputs to the cascade neural network should include: (1) current and previous state information $\{\nu_\xi,\ \nu_\eta,\ \dot{\theta}\}$, (2) previous output (command) information $\{\delta,\ P_f\}$, and (3) a description of the road visible from the current car position. More precisely, the network inputs are,

$$\{\nu_\xi(k - n_s), \cdots, \nu_\xi(k-1), \nu_\xi(k), \nu_\eta(k - n_s), \cdots, \nu_\eta(k-1), \nu_\eta(k), \dot{\theta}(k - n_s), \cdots, \dot{\theta}(k-1), \dot{\theta}(k)\} \quad (21)$$

$$\{\delta(k - n_c), \cdots, \delta(k-1), \delta(k), P_f(k - n_c), \cdots, P_f(k-1), P_f(k)\} \quad (22)$$

$$\{x(1), x(2), \cdots, x(n_r), y(1), y(2), \cdots, y(n_r)\} \quad (23)$$

where n_s is the length of the state histories and n_c is the length of the previous command histories presented to the network as input. For the road description, we partition the visible view of the road ahead into n_r equivalently spaced, body-relative (x, y) coordinates of the road median, and provide that sequence of coordinates as input to the network. Thus, the total number of inputs to the network n_i is,

$$n_i = 3n_s + 2n_c + 2n_r \quad (24)$$

The two outputs of the cascade network are $\{\delta(k+1), P_f(k+1)\}$. For the system as a whole, the cascade neural network can be viewed as a feedback controller, whose two outputs control the driving of the vehicle.

4 Performance criteria

Once we have abstracted models of driving control strategies from human control data, we would like to evaluate the skill or performance inherent in these models. Below, we review two such criteria for the task of human driving [5], which we will use subsequently in our performance optimization algorithm.

4.1 Obstacle avoidance

In real driving, obstacles such as rocks and debris can unexpectedly obstruct a vehicle's path and force the driver to react rapidly. Thus, obstacle avoidance is one important performance criterion by which we can gauge a model's performance. Since our HCS models receive only a description of the road ahead as input from the environment, we reformulate the task of obstacle avoidance as *virtual path following*. Assume that an obstacle appears a distance τ ahead of the driver's current position. Furthermore, assume that this obstacle completely obstructs the entire width of the road $(2w)$ and extends for a distance d along the road. Then, rather than follow the path of the actual road, we wish the HCS model to follow a virtual path as illustrated in Figure 2. This virtual path consists of (1) two arcs with

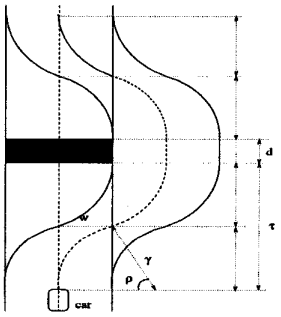

Figure 2: Virtual path for obstacle avoidance.

radius of curvature γ, which offset the road median laterally by $2w$, followed by (2) a straight-line segment of length d, and (3) another two arcs with radius of curvature γ which return the road median to the original path.

By analyzing the geometry of the virtual path, we can calculate the required radius of curvature γ of the virtual path segments as [5],

$$\gamma = \frac{\tau^2}{8w} + \frac{w}{2} \quad (25)$$

and the corresponding sweep angle ρ as,

$$\rho = sin^{-1}(\frac{\tau/2}{\gamma}) = sin^{-1}(\frac{\tau}{\frac{\tau^2}{4w} + w}) \quad (26)$$

Consider an obstacle located $\tau = 60$m ahead of the driver's current position. For this obstacle distance and $w = 5$m, γ evaluates to 92.5m. This is less than the minimum radius of curvature (100m) that we allow for the roads over which we collect our human control data. Hence, a particular HCS model may deviate significantly from the center of the road during the obstacle avoidance maneuver. In general, as the obstacle detection distance τ decreases, the maximum lateral offset increases [5]. Consequently, for a given model and initial velocity $v_{initial}$, there exists a value τ_{min} below which the maximum offset error will exceed the lane width w, We define the driving control for obstacle distances above τ_{min} to be stable; likewise, we define the driving control to be unstable for obstacle distances below τ_{min}

Now, we define the following obstacle avoidance performance criterion J_1:

$$J_1 = \frac{\tau_{min}}{v_{initial}} \quad (27)$$

where $v_{initial}$ is the velocity of the vehicle when the obstacle is first detected. The J_1 criterion measures to what extent a given HCS model can avoid an obstacle while still controlling the vehicle in a stable manner. The normalization by $v_{initial}$ is required, because slower speeds increase the amount of time a driver has to react and therefore avoiding obstacles becomes that much easier.

4.2 Tight turning

Here we analyze performance as a function of how well a particular HCS model is able to navigate tight turns. First, we define a special road connection consisting of two straight-line segments connected directly (without a transition arc segment) at an angle ζ. For small values of ζ, each HCS model will be able to successfully drive through the tight turn; for larger values of ζ, however, some models will fail to execute the turn properly by temporarily running off the road or losing complete sight of the road.

Now, define the maximum lateral offset error corresponding to a tight turn with angle ζ to be ψ. We can determine a functional relationship between ψ and ζ for a given HCS model. First, we take N measurements of ρ for different values of ζ where we denote the ith measurement as (ζ_i, ψ_i). Then, we assume a polynomial relationship between ψ and ζ such that,

$$\psi_i = \alpha_p \zeta_i^p + \alpha_{p-1} \zeta_i^{p-1} + \cdots + \alpha_1 \zeta_i + \alpha_0 + e_i \quad (28)$$

The least-squares estimate of the model ($\hat{\alpha}$) is given by,

$$\hat{\alpha} = (\hat{\zeta}^T \hat{\zeta})^{-1} \cdot \hat{\zeta}^T \cdot \hat{\psi} \tag{29}$$

where

$$\hat{\psi} = [\psi_1, \psi_2, \cdots \psi_N]^T \tag{30}$$

$$\hat{\zeta} = \begin{bmatrix} \zeta_1^p & \zeta_1^{p-1} & \cdots & \zeta_1 & 1 \\ \zeta_2^p & \zeta_2^{p-1} & \cdots & \zeta_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \zeta_N^p & \zeta_N^{p-1} & \cdots & \zeta_N & 1 \end{bmatrix}, \tag{31}$$

$$\hat{\alpha} = [\alpha_p, \alpha_{p-1}, \cdots, \alpha_0]^T \tag{32}$$

Previously, we have observed that the linear coefficient α_1 dominates the polynomial relationship in equation (28) [5]. Hence, as a first-order approximation, we define the following tight-turning performance criterion J_2 :

$$J_2 = \alpha_1 \tag{33}$$

5 Performance optimization

In section 4, we introduced two performance measures for evaluating the performance of our driving models. Below, we develop an algorithm for optimizing a learned control strategy model with respect to one of those (or for that matter, any other) performance criterion. There are two primary reasons why this may be necessary in order to successfully transfer control strategies from humans to robots.

First, while humans are in general very capable of demonstrating intelligent behaviors, they are far less capable of demonstrating those behaviors without occasional errors and random (noise) deviations from some nominal trajectory. The cascade learning algorithm will necessarily incorporate those in the learned HCS model, and will consequently be less than optimal. Second, control requirements may differ between humans and robots, where stringent power or force requirements often have to be met. Thus, a given individual's performance level may or may not be sufficient for a particular application.

Since a HCS model does offer an initially stable model, however, it represents a good starting point from which to further optimize performance. Let,

$$\omega = [w_1 \;\; w_2 \;\; \cdots \;\; w_n] \tag{34}$$

denotes a vector consisting of all the weights in the trained HCS model $\Gamma(\omega)$. Also let $J(\omega)$ denote any one performance criterion (e.g. J_1 or J_2 in the previous section). We would now like to determine the weight vector ω^* which optimizes the performance criterion $J(\omega)$. This optimization is difficult in principle because (1) we have no explicit gradient information

$$G(\omega) = \frac{\partial}{\partial \omega} J(\omega) \tag{35}$$

and (2) each experimental measurement of $J(\omega)$ requires a significant amount of computation. We lack explicit gradient information, since we can only compute our performance measures empirically. Hence, gradient-based optimization techniques, such as steepest descent and Newton-Raphson are not suitable. And because each performance measure evaluation is potentially computationally expensive, genetic optimization, which can requires many iterations to converge, also does not offer a good alternative. Therefore we turn to *simultaneously perturbed stochastic approximation (SPSA)* to carry out the performance optimization.

Stochastic approximation (SA) is a well known iterative algorithm for finding roots of equations in the presence of noisy measurements. Simultaneously perturbed stochastic approximation (SPSA) [7] is a particular multivariate SA technique which requires as few as two measurements per iteration and shows fast convergence in practice. Hence, it is well suited for our application. Denote ω_k as our estimate of ω^* at the kth iteration of the SA algorithm, and let ω_k be defined by the following recursive relationship:

$$\omega_{k+1} = \omega_k - \alpha_k \bar{G}_k \tag{36}$$

where $\bar{G}_k$ is the simultaneously perturbed gradient approximation at the kth iteration,

$$\bar{G}_k = \frac{1}{p} \sum_{i=1}^{p} G_k^i \approx \frac{\partial}{\partial \omega} J(\omega) \tag{37}$$

$$G_k^i = \frac{J_k^{(+)} - J_k^{(-)}}{2c_k} \begin{bmatrix} 1/\Delta_{kw_1} \\ 1/\Delta_{kw_2} \\ \cdots \\ 1/\Delta_{kw_n} \end{bmatrix} \tag{38}$$

Equation (37) averages p stochastic two-point measurements G_k^i for a better overall gradient approximation, where,

$$J_k^{(+)} = J(\omega_k + c_k \Delta_k) \tag{39}$$

$$J_k^{(-)} = J(\omega_k - c_k \Delta_k) \tag{40}$$

$$\Delta_k = [\Delta_{kw_1} \Delta_{kw_2} \cdots \Delta_{kw_n}]^T \tag{41}$$

and where Δ_k is a vector of mutually independent, mean-zero random variables (e.g. symmetric Bernoulli distributed), the sequence $\{\Delta_k\}$ is independent and identically distributed, and the $\{\alpha_k\}$, $\{c_k\}$ are positive scalar sequences satisfying the following properties:

$$\alpha_k \to 0, \quad c_k \to 0 \; as \; k \to \infty, \tag{42}$$

$$\sum_{k=0}^{\infty} \alpha_k = \infty, \quad \sum_{k=0}^{\infty} (\frac{\alpha_k}{c_k})^2 < \infty \tag{43}$$

The weight vector ω_0 is of course the weight representation in the initially stable learned cascade model. Larger values of p in equation (37) will give more accurate approximations of the gradient. Figure 3 illustrates the overall performance-optimization algorithm.

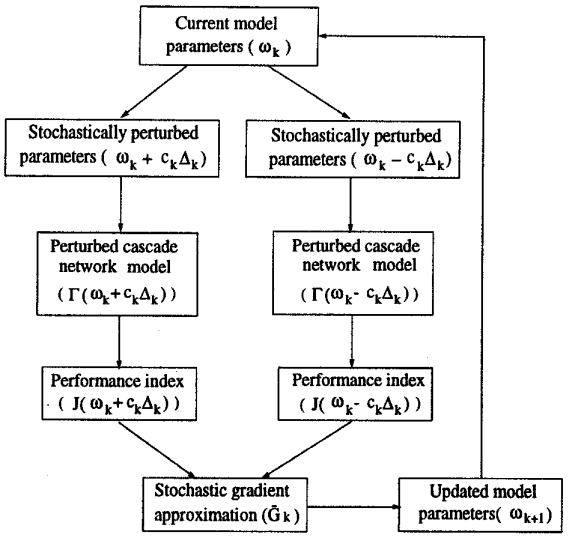

Figure 3: Stochastic optimization algorithm.

6 Experiment

6.1 Results

Here, we test the performance optimization algorithm on control data collected from two individuals, Harry and Dick. In order to simplify the problem somewhat, we keep the applied force constant at $P_f = 300N$. Hence, the user is asked to control only the steering δ.

For each person, we train a two-hidden-unit HCS model with $n_s = n_c = 3$, and $n_r = 15$; because we are keeping P_f constant, the total number of inputs for the neural network models is therefore $n_i = 42$.

Now, we would like to improve the tight-turning performance criterion J_2 defined in equation (33) for each of the trained models. In the SPSA algorithm, we empirically determine the following values for the scaling sequences $\{\alpha_k\}$, $\{c_k\}$:

$$\alpha_k = 0.000001/k, \quad k > 0 \tag{44}$$

$$c_k = 0.001/k^{0.25}, \quad k > 0 \tag{45}$$

We also set the number of measurements per gradient approximation in equation 37 to $p = 1$. Finally, denote J_2^k as the criterion J_2 after iteration k of the optimization algorithm; hence, J_2^0 denotes the performance measure prior to any optimization.

Figure 4 plots $100 \times J_2^k/J_2^0$, $0 \leq k \leq 60$, for the HCS models corresponding to Dick and Harry. We note that for Dick, the performance index J_2 improves from $J_2^0 = 25.5$ to $J_2^{60} = 12.5$. For Harry, the improvement is less dramatic; his model's performance index improve from $J_2^0 = 17.7$ to $J_2^{60} = 16.1$. Thus, the performance optimization algorithm is able to improve the performance of Dick's model by about 55% and Harry's model by about 9% over their respective initial models. In other words, the optimized models are better for negotiating tight turns without running off the road.

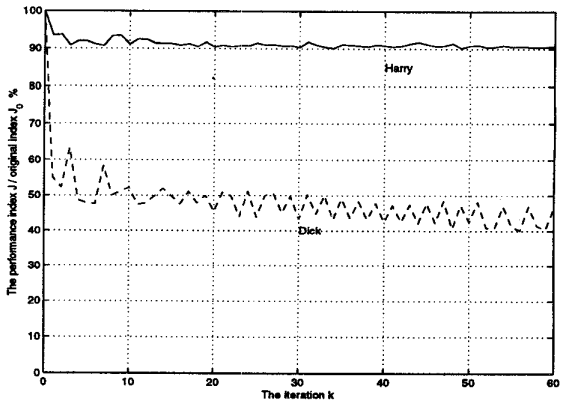

Figure 4: Performance improvement in stochastic optimization algorithm.

From Figure 4, we observe that most of the improvement in the optimization algorithm occurs in the first few iterations. Then, as $k \to \infty$, J_2^k converges to a stable value since $\alpha_k, c_k \to 0$. Clearly, the extent to which we can improve the performance in the trained HCS models depends on the characteristics of the original models. Dick's initial performance index of $J_2^0 = 25.5$ is much worse than Harry's initial performance index of $J_2^0 = 17.7$. Therefore, we would expect that Dick's initial model lies further away from the nearest local minimum, while Harry's model lies closer to that local minimum. As a result, Harry's model can be improved only a little, while Dick's model has much larger room for improvement.

6.2 Discussion

Below we discuss some further issues related to performance optimization, including (1) the effect of performance optimization on other performance criteria, and (2) the similarity of control strategies before and after performance optimization.

First, we show how performance improvement with respect to one criterion can potentially affect performance improvement with respect to a different criterion. Consider Dick's HCS model once again. As we have already observed, his tight turning performance criterion improves from $J_2^0 = 25.5$ to $J_2^{60} = 12.5$. Now, let J_1^0 denote the obstacle avoidance performance criterion for Dick's initial HCS model, and let J_1^{60} denote the obstacle avoidance performance criterion for Dick's HCS model, optimized with respect to J_2. Figure 5 plots the maximum offset from the road median as a function of the obstacle detection distance τ for Dick's initial model (solid line) and Dick's optimized model (dashed line), where $v_{initial} = 35$.

From Figure 5, we can calculate J_1^0 and J_1^{60}:

$$J_1^0 \approx \frac{42}{35} = 1.20 \tag{46}$$

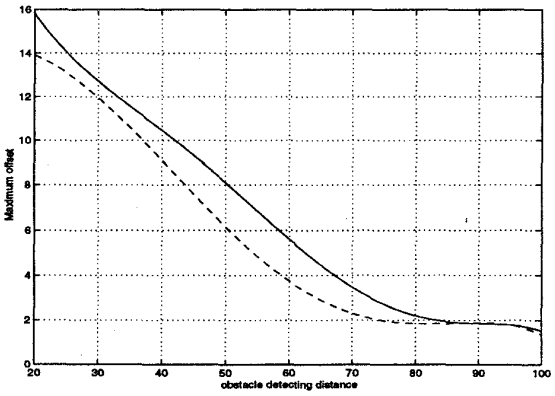

Figure 5: Maximum lateral offset for original (solid) and final (dashed) HCS models.

$$J_1^{60} \approx \frac{36}{35} = 1.03 \tag{47}$$

Thus, Dick's optimized HCS model not only improves tight turning performance, but obstacle-avoidance performance as well. This should not be too surprising, since the tight turning and obstacle avoidance behaviors are in fact tightly related. During the obstacle avoidance maneuver, tight turns are precisely what is required for successful execution of the maneuver.

Second, we would like to see how much performance optimization changes the model's control strategy away from the original human control approach. To do this we turn to a hidden Markov model-based similarity measure [4] developed for comparing human-based control strategies. Let H_x denote the human control trajectory for individual x, let M_x denote control trajectories for the unoptimized model corresponding to individual x, and let O_x denote control trajectories for the optimized model (with respect to J_2) corresponding to individual x. Also let $0 \leq \sigma(A,B) \leq 1$ denote the similarity measure for two different control trajectories A and B, where larger values indicate greater similarity, while smaller values indicates greater dissimilarity between A and B.

For each individual, we can calculate the following three similarity measures:

$$\sigma(H_x, M_x) \tag{48}$$
$$\sigma(H_x, O_x) \tag{49}$$
$$\sigma(M_x, O_x) \tag{50}$$

Table 1 lists these similarities for Dick and Harry.

	$x = Dick$	$x = Harry$
$\sigma(H_x, M_x)$	0.762	0.573
$\sigma(H_x, O_x)$	0.434	0.469
$\sigma(M_x, O_x)$	0.544	0.823

Table 1: Control strategy similarity

From our experience with this similarity measure, we note that all the values in Table 1 indicate significant similarity. Specifically, the similarities for $\sigma(H_x, O_x)$ (0.434 and 0.469) suggest that even after performance optimization, a substantial part of the original human control strategy is preserved. Furthermore, the other similarity measures are consistent with the degree of performance improvement in each case. For Dick, where a substantial performance improvement of 55% was achieved, the similarity between the initial and optimized models is far less than Harry, where the performance improvement was more incremental.

7 Conclusion

In this paper, we have proposed an iterative optimization algorithm, based on simultaneously perturbed stochastic approximation (SPSA), for improving the performance of learned models of human control strategy. The algorithm keeps the overall structure of the learned models in tact, but tunes the parameters (i.e. weights) in the model to achieve better performance. It requires no analytic formulation of performance, only two experimental measurements of a defined performance criterion per iteration. We have demonstrated the viability of the approach for the task of human driving, where we model the humans control strategy through cascade neural networks. While performance improvements vary between HCS models, the optimization algorithm always settles to stable, improved performance after only a few iterations. Furthermore, the optimized models retain important characteristics of the original human control strategy.

Acknowledgments

This work is supported in part by Hong Kong Research Council Grant No. CUHK4138/97E and Carnegie Mellon University.

References

[1] M. Sugeno and T. Yasukawa, "A fuzzy-Logic-Based Approach to Qualitative Modeling," *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 1, 1993

[2] U. Kramer, "On the Application of Fuzzy Sets to the Analysis of the System Driver-Vehicle-Environment," *Automatica*, vol. 21, no. 1, pp. 101-7, 1985.

[3] M. C. Nechyba and Y. Xu, "Human Control Strategy: Abstraction, Verification and Replication," *IEEE Control Systems Magazine*, vol. 17, no. 5, pp. 48-61, 1997.

[4] M. C. Nechyba and Y. Xu, "Stochastic Similarity for Validating Human Control Strategy Models," *Proc. IEEE Conf. on Robotics and Automation*, vol. 1. pp. 278-83, 1997.

[5] J. Song, Y. Xu, M. C. Nechyba and Y. Yam, "Two Measures for Evaluating Human Control Strategy," *Proc. IEEE Conf. on Robotics and Automation*, vol. 3. pp. 2250-55, 1998.

[6] M. C. Nechyba and Y. Xu, "Cascade Neural Networks with Node-Decoupled Extended Kalman Filtering," *Proc. IEEE Int. Symp. on Computational Intelligence in Robotics and Automation*, vol. 1, pp. 214-9,1997.

[7] J. C. Spall, " Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation," *IEEE Trans. Automation Control*, vol. 37, no. 3, pp. 332-41, 1992.