Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Driver Modeling for Teleoperation with Time Delay

Steve Vozar [*] Dawn M. Tilbury [*]

[*] *Mechanical Engineering Department, University of Michigan, Ann Arbor, MI 48108-2125 USA (e-mail: {svozar,tilbury}@umich.edu).*

Abstract: The results from a user study on teleoperated steering tasks are used to create a driver model under different latency conditions. The 31-subject user study explored the effects of latency on teleoperated steering tasks using a simulated mobile robot receiving input commands from a teleoperator via a computer gamepad. Using fundamental concepts from automotive steering models, and examining the users' input commands to the simulated robot under different latency conditions, a model of a human teleoperator for steering tasks was developed, tuned and validated. The model is a PD controller with feedback based on the projected lateral displacement of the robot. The tuning of the model gains for different latency scenarios reflects the real-world control strategies that users must employ when adapting to system latency. Simulation results show that the control gains can be interpolated to predict teleoperator performance under latency scenarios that were not tested with users. An analysis of the closed-loop stability of the system confirms our empirical observation that the ratio of the controller gains K_d/K_p increases as the latency increases.

Keywords: Teleoperation, Mobile robots, Human-Machine Interface, Driver models, Time delay.

1. INTRODUCTION

Latency is a significant factor affecting teleoperated robot performance. Whether the latency originates in the system communications network, processing routines, or sensing hardware, it can negatively impact a human operator's ability to perform even basic remote tasks. With enough delay, a user's entire control strategy is often switched to a move-and-wait open-loop methodology (Sheridan and Ferrell, 1963), making it impossible to maneuver a robot quickly or efficiently. Moreover, while operators can sometimes adapt to a system delay if it remains relatively consistent, variable latency makes it difficult for humans to predict how the robot will respond (Luck *et al.*, 2006; Davis *et al.*, 2010). While many teleoperated industrial or surgical robots have the benefit of communicating over wired networks, mobile robots generally must utilize wireless protocols, which have higher latency and latency variation.

Understanding how teleoperators interact with robots is key to designing better teleoperation systems. Because it is often not feasible to test large numbers of different design iterations with real human operators, it is desirable to have a model of a human teleoperator that could be used to evaluate multiple designs quickly and easily. Driver models used for similar purposes have a long history in the automotive industry (MacAdam, 2003), but these models may not be directly applicable to teleoperation, as the two tasks are quite different. While vehicle drivers have a wide variety of sensory feedback available, teleoperators are generally limited to relying solely on visual feedback, which is often delayed and has a limited field of view (Chen *et al.*, 2007). Additionally, input devices for automobiles (*e.g.* steering wheels with haptic feedback) are generally not the same as those used in teleoperation, which can be as simple as off-the shelf video game controllers. Finally, the internal models automobile drivers have for their vehicles are likely far more developed than those of even experienced teleoperators. Thus, there is a need develop new models of humans performing teleoperation tasks in remote environments.

The results in this paper are based on a 31-subject user study designed to measure the effects of latency on a simulated teleoperation steering task using a commercially available gamepad as an input device. The input commands from the human to the robot were recorded with the aim of developing a driver model to simulate human behavior for simple steering tasks under different latency conditions. The results in this paper demonstrate that, for this particular scenario, a teleoperator's steering commands with a gamepad can be reasonably modeled as PD controller based on the preview of the robot's anticipated lateral displacement. To the our knowledge, this is the first steering model developed specifically for teleoperated robots.

The paper is organized as follows: First, prior work is discussed in Section 2. Then Section 3 describes the user study. The driver model is developed and validated using the test data in Section 4, and the closed-loop stability of the system is analyzed in Section 5. Finally, Section 6 discusses conclusions and future work regarding the direction of this research.

2. BACKGROUND

2.1 Latency in Teleoperation

It is well-established that latency has a detrimental impact on teleoperation performance, and time delay is known to be one of the most significant factors affecting remote perception and manipulation (Chen *et al.*, 2007). Sources of latency in a teleoperated robot system include network delays, sensing delays, and processing delays, as well as delays caused by the operator's cognitive and physical processing, which can themselves be affected by the delay in the rest of the feedback loop (Vozar, 2013).

⋆ This research was supported in part by the Automotive Research Center (ARC) at the University of Michigan, with funding from government contract DoD-DoA W56H2V-04-2-0001 through the US Army Tank Automotive Research, Development, and Engineering Center.

One of the earliest studies in this domain investigated open-loop position control of a remote manipulator, and found that users adopted a move-and-wait strategy when the delay was above 1 second (Sheridan and Ferrell, 1963). More recent studies have examined mobile robot teleoperation performance under other conditions including 2D driving (Luck *et al.*, 2006; Davis *et al.*, 2010) and 3D underwater navigation tasks (Corde Lane *et al.*, 2002). Prior work has shown that variable latency leads to worse performance than constant latency, because users are less able to compensate for the changing delays (Luck *et al.*, 2006; Davis *et al.*, 2010). The directionality of the latency (whether user-to-robot or robot-to-user) has also been investigated, where it has been found that users felt robot control was more difficult when the latency was in the robot-to-user direction, but no objective difference in performance was observed (Luck *et al.*, 2006).

2.2 *Driver Models*

Modeling human driver behavior has a rich history in the automotive domain (MacAdam, 2003; Ungoren and Peng, 2005; Delice and Ertugrul, 2007). From transfer function models to nonlinear and adaptive controllers to neural networks, genetic algorithms, and fuzzy logic controllers (MacAdam, 2003; Delice and Ertugrul, 2007), there are myriad methodologies for modeling vehicle lateral control (steering), longitudinal control (acceleration and braking), and combined control. These models can be used to simulate human drivers when testing new vehicle designs and technologies (MacAdam, 2003; Delice and Ertugrul, 2007), and despite the complexity of human behavior, low order models are often sufficient for many control tasks (Brito Palma *et al.*, 2012).

Regardless of overarching approach, all driver models aim to capture the key characteristics of the human driver as a controller in a feedback loop. MacAdam (2003) notes that the essential requirements of a model should include: a time delay due to human processing, a preview of the upcoming control requirements, the ability to adapt to different vehicle and operating conditions, and an internal model to predict vehicle responses. Our modeling efforts adhere to these requirements.

Automotive steering models can use one or more feedback cues as inputs to the driver model, including any or all of the following: lateral displacement, lateral acceleration, roll angle, heading angle, and yaw rate. The cues can be processed by the model in one or more forms, which may include visual cues, motion effects, sound, and tactile information (MacAdam, 2003). However, typically only limited visual signals are available in teleoperation scenarios.

3. USER STUDY SETUP

A user study was performed to gather data on task performance and operator driving style in robot steering tasks using a low-fidelity simulation of a teleoperated mobile robot driving on a simulated test track of constant width (see Fig. 1). The user's goal was to steer the robot such that it followed the track's indicated centerline as closely as possible.

3.1 *Simulation Environment*

A custom simulation environment for this set of tests (with a look inspired by the classic arcade game *Pole Position*) was written in Java using the April Robotics Toolkit (APRIL Robotics Laboratory, 2012) and Lightweight Communications

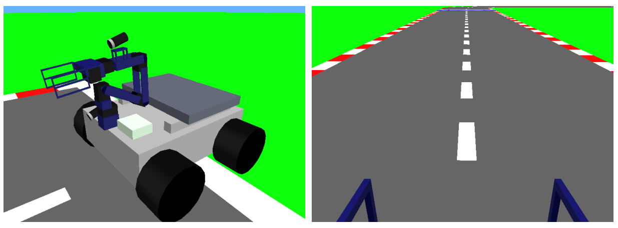

Fig. 1. Renderings of the simulated robot, and the first-person viewpoint presented to the users in the study.

and Marshalling (LCM) (Huang *et al.*, 2010) libraries. The simulation uses a simple kinematic driving model of a representative skid-steer robot chassis, calculating the robot's Cartesian position (x_1,x_2) and orientation (θ) for each time step ($k+1$) using the commanded robot speed (v) and turn rate (ω):

$$\begin{bmatrix} x_1 \\ x_2 \\ \theta \end{bmatrix}_{k+1} = \begin{bmatrix} x_1 \\ x_2 \\ \theta \end{bmatrix}_{k} + \begin{bmatrix} v_k \cos\theta_{k+1}\Delta t \\ v_k \sin\theta_{k+1}\Delta t \\ \omega_k \Delta t \end{bmatrix} \quad (1)$$

The simulation runs in a Java thread at a rate of 60Hz.

3.2 *User Interface*

The user gives steering commands to the robot using one of the mini joysticks of a computer gamepad (Logitech Cordless Rumblepad 2), which is read by the Operator Control Unit (OCU) at a rate of 40Hz. A small amount of noise is artificially added the gamepad input, generated from a uniform distribution on the range of [-10%,10%] of the maximum possible input command. Operators do not have control over the robot speed; it is constant for the duration of the trial unless the robot is driven off of the track. The simulation visualization is displayed to the user via the OCU on a 25" (63.5cm) monitor with a resolution of 1920x1200px in a full-screen window. The visualization refreshes the displayed frame at a rate of 15Hz.

3.3 *Insertion of Delay*

Gamepad instruction packets are read by the OCU and enter a queue waiting to be read by the simulation, representing a delay in the human-to-robot direction. For each incoming instruction, a simulated delay (δ) is inserted between the gamepad instruction and the simulation.

This induced delay is added to the system and does not include or compensate for any further computer processing delays or delays due to the display device, which are assumed to be negligible compared to the magnitude of the latencies induced in the trials. Additionally, there is a slight delay from the sampling time of the gamepad, which is also negligible.

3.4 *Test Track*

Sixteen non-intersecting test tracks were generated that each contain exactly one of the following elements: Right Turn, Left Turn, U-Turn Right, U-Turn Left, S-Turn Right-Left, and S-Turn Left-Right. All turns have a constant radius of 2m, and the width of the track is 2m, with 0.125m borders on either side. The width of the robot (wheel-to-wheel) is 0.74m. The turn gain β of the gamepad input is determined from the robot speed such that the minimum turning radius of the robot is 1.6m, preventing users from relying on the actuator limits of the gamepad to execute ideal turning motions. Here, $\beta = 0.6375$.

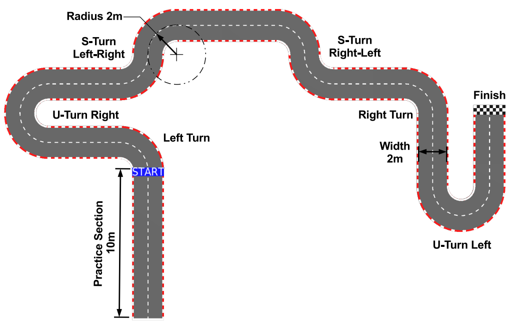

Fig. 2. Representative track with dimensions for simulated driving tasks.

Each track element has a section of straight-line path at least 5m long immediately following it to allow the user to try to recover from any deviation sustained during the turn. Additionally, there is a 10m straight-line practice section at the start of each track. A sample track is shown in Fig. 2.

3.5 Scoring

The path-following score for each trial is determined as a function of the robot's distance from the centerline over the course of the path. Scoring begins after the robot has passed the start line indicating the end of the practice section of track. The score at time step i is given by:

$$S_i = \max(0, 1 - |y_i|) \tag{2}$$

where y_i is the lateral displacement at step i. The total score is determined as the average of the scores at each time step:

$$S = \frac{1}{n}\sum_{i=1}^{n} S_i \tag{3}$$

Therefore, a score of 1 indicates that the path was followed perfectly, and a score of 0 indicates that the robot was never on the test track.

3.6 Procedure

User tests were conducted with 32 volunteers recruited via flier and email advertisement from a population of engineering undergraduate and graduate students. One participant withdrew from the study, leaving 31 users in the data set. A total of 22 men and 9 women completed the tasks, ranging in ages from 18 to 37, with a mean age of 23.5 years (standard deviation = 4.2 years). Users were given \$10 for participating, with the knowledge that an additional \$10 bonus would be awarded to the top performers as determined at the end of the trials, with a \$30 bonus cap. The tests were designed to take less than one hour, and most participants needed approximately 45 minutes. These tests were approved by the University of Michigan Health Sciences and Behavioral Sciences Institutional Review Board. (UM IRB #HUM00044265). The tests included different viewpoints of the robot (first and third person), different robot speeds (1.0 m/s and 1.5 m/s) and different delays (both constant and variable). The order of the scenarios was randomized to counterbalance any learning or fatigue effects. Additionally, the order of the speeds in each scenario was randomized, as was the selection of track for each trial. No significant differences were found between the first- and third-person viewpoints. In this paper, we consider only the 1.0 m/s robot speed and constant delays.

Table 1. Latencies introduced in the user tests.

Latency Type	δ
A	0
B	250 ms
C	500 ms
D	750 ms

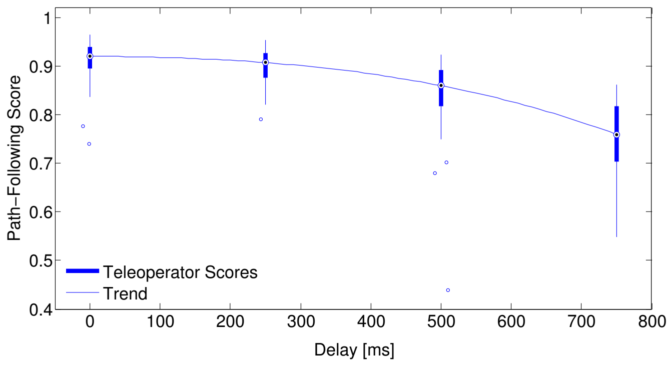

Fig. 3. Model of user performance as measured by the path-following score for various latencies and robot speeds.

4. DRIVER MODEL DEVELOPMENT

This section discusses the development of a model for simulating the steering commands issued by the teleoperator under different system latency conditions. This model could be useful as a substitute for a real teleoperator in simulations of mobile robot tasks requiring steering inputs.

4.1 Driver performance results

Figure 3 shows the score distribution for the constant latency scenarios plotted with a trend line through the median values, for the robot speed 1.0 m/s, with which we can predict path-following scores for delays not explicitly tested in this study.

4.2 Driver Behavior

To act as an acceptable substitute for a human driver, the steering model must accurately replicate the key characteristics of the human driver. Figure 4 shows two example datasets from runs under low latency (Latency A), and under high latency (Latency D), which are representative of the test datasets.

We would like our driver model to emulate the characteristics of the input command. As shown in both traces in Fig. 4, the input command is almost always saturated. This is because most users tended to move the control stick on the gamepad as far to the left or right as its travel allowed rather than use an intermediate input. This tendency was present under all test conditions: over all the trials, users kept the joystick centered 55.5% of the time, pushed the stick to the far left or right 33.5% of the time, and only 11% of commands were any value in between. This type of gamepad input behavior has been previously observed in computer racing games (Brown *et al.*, 2010). Our users used both quick, frequent adjustments, and long sustained turning commands.

4.3 Model Development

To develop a model for teleoperated robot steering, we can draw from some of the techniques previously developed for automotive steering models. Specifically, we use a preview of the desired path combined with an internal model of the vehicle kinematics and an operator time delay.

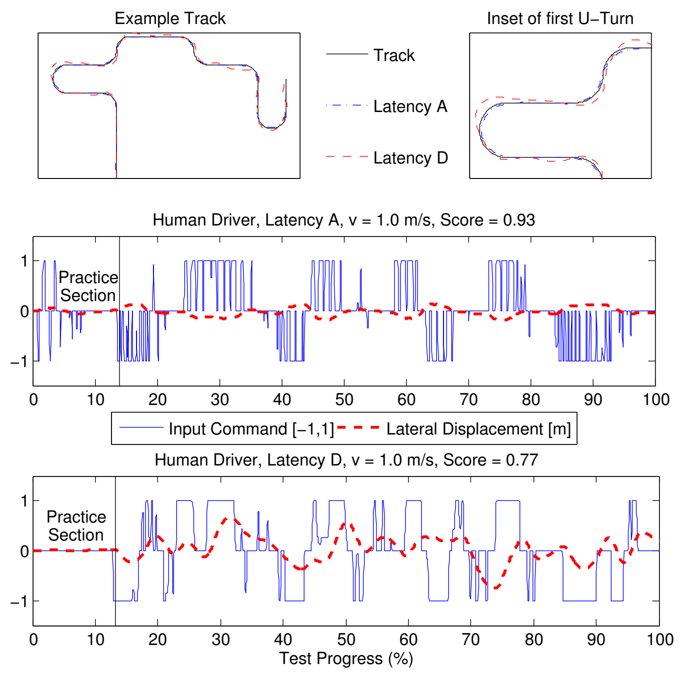

Fig. 4. Plots showing example datasets of low-latency and high-latency test cases (from two different users).

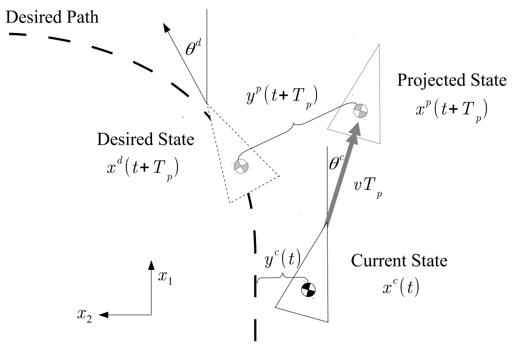

Fig. 5. Diagram illustrating the determination of the projected lateral displacement.

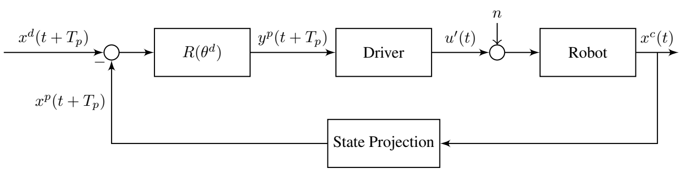

Fig. 6. Steering control loop. Lateral displacement $y^p(t+T_p)$ is determined from the difference between the projected and desired robot states. The $R(\theta^d)$ block represents the rotation operation described in Eq. 6. The n term represents the noise injected into the command signal.

A simple steering model can be developed based on the driver's anticipated deviation from the desired path at some future time $(t + T_p)$. In this case, we choose the projected lateral displacement $y^p(t+T_p)$ of the robot as the feedback cue. Figure 5 illustrates the process of calculating the projected lateral displacement, which is based on the projected state of the robot $x^p(t+T_p)$, assuming it continues its trajectory from the current state $x^c(t)$ at a constant velocity:

$$x^p(t+T_p) = \begin{bmatrix} x_1^p \\ x_2^p \\ \theta^p \end{bmatrix} = \begin{bmatrix} x_1^c \\ x_2^c \\ \theta^c \end{bmatrix} + \begin{bmatrix} v\cos\theta^c \\ v\sin\theta^c \\ 0 \end{bmatrix} T_p \quad (4)$$

The desired future state of the robot $x^d(t+T_p)$, is the point along the desired path closest to the projected state, as measured by Euclidian distance:

$$x^d(t+T_p) = \underset{x\in path}{\arg\min}\sqrt{(x_1 - x_1^p)^2 + (x_2 - x_2^p)^2} \quad (5)$$

The projected lateral displacement is the component of the difference between the projected and desired states perpendicular to the direction of the desired path. It is obtained by rotating the vector from x^d to x^p by the desired heading angle θ^d and taking the component perpendicular to the path:

$$y^p(t+T_p) = (x_2^d - x_2^p)\cos\theta^d - (x_1^d - x_1^p)\sin\theta^d \quad (6)$$

For a continuous path, this is equivalent to taking the length of the vector, but for paths consisting of discrete points this method results in smaller errors due to gaps in the path.

We now model the steering action of the user as a PD controller (Nise, 2004) based on the anticipated lateral displacement feedback cue, with an additional delay δ_H representing the driver's physical reaction:

$$u(t+\delta_H) = K_p y^p(t+T_p) + K_d \dot{y}^p(t+T_p) \quad (7)$$

The control signal generated by Eq. 7 is continuous and unbounded. However, the gamepad input device is only capable of generating control inputs on the interval [-1, 1], and it was noted in Section 4.2 that users tend to issue commands at one extreme of the interval or the other. We can capture both the actuator saturation and the users' tendency to max out the limits of the gamepad by conditioning the control input with a simple threshold ($\mu > 0$):

$$u'(t) = \begin{cases} -1 & \text{if } \quad u(t) \le -\mu \\ 0 & \text{if } \mu > u(t) > -\mu \\ 1 & \text{if } \mu \le u(t) \end{cases} \quad (8)$$

and using $u'(t)$ as the simulated gamepad steering command issued to the robot. Figure 6 shows a block diagram of the overall steering control loop.

4.4 Model Parameter Tuning

We focus here on the case in which the robot speed is 1 m/s. To simplify the process of tuning of this model to reflect the driver behaviors measured in the user tests, we can make some assumptions about the parameters. First, we assume that the physical reaction time (δ_H) of the model driver is 200ms, as the gamepad log data indicates users actuated the joystick at approximately the same time scale. Also, we assume a lookahead time (T_p) of 1250ms. Additionally, we set the threshold for conditioning the control input to $\mu = 0.5$. Therefore the only two parameters left to tune are the control gains K_p and K_d. The gains were tuned by hand to reflect the path-following score and average control input of the users, discussed in Section 4.2, for each constant latency case, using a MATLAB model of the robot system in place of the Java simulation. A summary of these constant and tuned parameters is shown in Table 2.

For the zero latency case, the K_d value of zero is consistent with vehicle steering models having only proportional feedback to errors in projected lateral displacement (Toffin *et al.*,

Table 2. Tuned control gains and parameter values for different latency cases.

Latency	δ [ms]	K_p	K_d	T_p [ms]	δ_H [ms]	μ
A	0	1.7	0.0	1250	200	0.5
B	250	1.6	0.3	1250	200	0.5
C	500	1.3	0.7	1250	200	0.5
D	750	1.0	1.0	1250	200	0.5

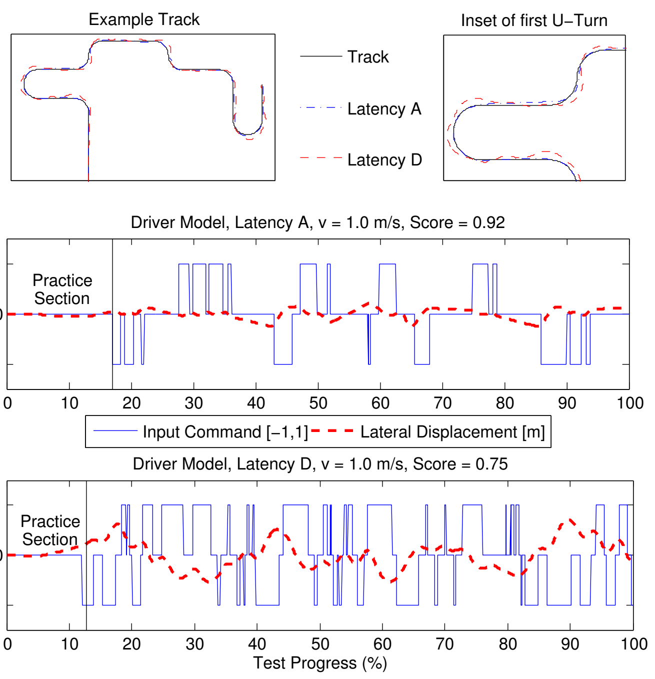

Fig. 7. Example paths of the robot using the steering model, similar to those produced by human drivers.

2007). With latency, the ratio of K_d/K_p increases as the latency increases, demonstrating that the steering model more heavily weighs the projected error for low latency, and relies more on the predicted displacement trend when the delay is high. Intuitively, this reflects the strategy employed by a human teleoperator in the control loop, who must rely more on prediction based on anticipation of the track's features when the latency is high rather than direct visual feedback. Additionally, the decreasing proportional gain reflects the users' increased tolerance for steady state errors in the difficult-to-control high latency cases.

Because the noise n injected into the input command propagates through the robot system, the $\dot{y}^p(t+T_p)$ term can also be quite noisy, significantly affecting the derivative portion of the controller. Therefore, the derivative term is smoothed by averaging the values of $\dot{y}^p(t+T_p)$ over 10 samples.

4.5 Model Validation

To test the performance of the steering model, the teleoperation scenarios were run with the gamepad command simulated in real-time by a MATLAB script running the PD steering model at 40Hz and communicating to the robot simulation via LCM over the gamepad channel. Everything else about the simulation was the same as the setup with the human operator. Each of the latency scenarios A-D were run five times on five different test tracks with a robot speed of 1 m/s. In addition, two intervening constant latencies were tested, again with five trials each, at $\delta = 380, 660$ms. For these intermediate cases, gain values K_p and K_d were determined by linearly interpolating between the values in Table 2.

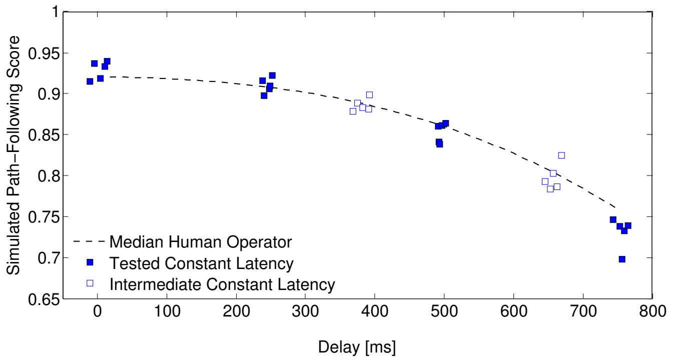

Fig. 8. Scores of path-following simulations of the robot at a speed of 1 m/s using the steering model.

Figure 7 shows two example datasets generated by the steering model, which appear similar to the datasets produced by the human drivers shown in Fig. 4. Both the saturated input behavior and the overall lateral displacement profiles are qualitatively captured by the steering model.

As shown in Fig. 8, the steering model is able to emulate the median path following scores of the of the operators in the user study. Additionally, the simulations at intermediate constant latency values not explicitly measured in the user trials follow the trend of the measured data, indicating that the gains determined by interpolation are acceptable.

5. STABILITY ANALYSIS

Using the PD controller model to represent the user, the stability of the closed-loop in Fig. 6 can be analyzed under different delays. For this analysis, we consider straight-line motion, and assume small angles ($\cos\theta \approx 1, \sin\theta \approx \theta$). In this case, the robot equations simplify to:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}v\\ v\theta\\ \omega\end{bmatrix} \tag{9}$$

The projected state is:

$$x^p(t+T_p) = \begin{bmatrix}x_1^c + vT_p\\ x_2^c + v\theta T_p\\ \theta^c\end{bmatrix} \tag{10}$$

For a straight line, the desired state is:

$$x^d(t+T_p) = \begin{bmatrix}x_1^d\\ 0\end{bmatrix} \tag{11}$$

and the desired angle is $\theta^d = 0$, therefore the offset is

$$\begin{aligned} y^p(t+T_p) &= (x_2^d - x_2^p)\cos\theta^d - (x_1^d - x_1^p)\sin\theta^d && (12)\\ &= -x_2^p = -x_2^c + v\theta T_p && (13)\end{aligned}$$

Using state-space notation, with input $u = \omega$ and reference $r(t)$, we have

$$\begin{bmatrix}\dot{x}^c\\ \dot{\theta}^c\end{bmatrix} = \begin{bmatrix}0 & v\\ 0 & 0\end{bmatrix}\begin{bmatrix}x^c\\ \theta^c\end{bmatrix} + \begin{bmatrix}0\\ \beta\end{bmatrix} u = Az + Bu \tag{14}$$

$$y^p = -\begin{bmatrix}1 & vT_p\end{bmatrix}\begin{bmatrix}x^c\\ \theta^c\end{bmatrix} = Cz \tag{15}$$

Adding in the PD controller, and a delay δ between the input and the robot, we can compute:

Table 3. Closed-loop poles with driver models at different latencies.

Latency Type	δ	Closed-loop poles	
A	0	$-0.68 \pm 0.79j$	
B	250 ms	$-0.66 \pm 0.81j$	-9.8
C	500 ms	$-0.56 \pm 0.68j$	-9.6
D	750 ms	$-0.49 \pm 0.55j$	-15.6

$$sZ(s) = AZ(s) + BU(s) = AZ(s) + BK(s)E(s) \quad (16)$$
$$= AZ(s) + BK(s)\left(R(s) - Y^p(s)\right) \quad (17)$$
$$\left(sI - A + BK(s)C\right)Z(s) = BK(s)R(s)$$

and the characteristic polynomial can be found:

$$s(s + vT_p\beta K(s)) + v\beta K(s)$$

where $K(s)$ is the product of the PD controller and the delay. Using a first-order Padé approximation for the delay δ, we get

$$K(s) = (K_p + K_d s)\left(\frac{1 - \delta s/2}{1 + \delta s/2}\right)$$

A necessary condition for stability is that all of the coefficients in the characteristic polynomial are positive. Defining $m = v\beta(T_pK_p + K_d)$, and performing some algebra, we get the set of conditions:

$$vT_p\beta K_d < 1 \quad (18)$$
$$\delta m < 2(1 + vT_p\beta K_d) < 4 \quad (19)$$
$$\delta < 2\left(T_p + \frac{K_d}{K_p}\right) \quad (20)$$

These conditions show that the maximum delay that can be tolerated depends on the robot speed v, the turn gain β, the lookahead distance T_p and the control gains K_p, K_d. In particular, condition (20) agrees with our observation from tuning the gains that the ratio of K_d/K_p increases as the latency increases.

For small delay, the dominant closed-loop poles are similar to those of the no-delay case. See Table 3.

6. CONCLUSIONS

This paper presents the results of a 31-subject user study exploring the effects of latency on teleoperated steering tasks using a simulated mobile robot receiving input commands from a teleoperator via a computer gamepad. A model of user performance under constant latency was developed, and is shown in Fig. 3.

Using the fundamental concepts from automotive steering models, and examining the users' input commands to the simulated robot under different latency conditions, a model of a human teleoperator for steering tasks was developed, tuned and validated. The model is a PD controller with feedback based on the projected lateral displacement of the robot. The tuning of the model gains for different latency scenarios reflects the real-world control strategies that users must employ when adapting to system latency. Simulation results show that the control gains can be interpolated to predict teleoperator performance under latency scenarios that were not tested with users. An analysis of the closed-loop stability of the system confirms our empirical observation that the ratio of the controller gains K_d/K_p increases as the latency increases.

Our current work is exploring the effects of variable latency in teleoperation scenarios. We are also considering higher-fidelity simulation models, and comparing the results of user tests in simulation to user tests with physical hardware.

One natural extension of this work is the development of longitudinal and/or combined driver models for teleoperated mobile robots. It may also be possible to similarly model teleoperators in other contexts, such as pointing or object manipulation tasks.

ACKNOWLEDGEMENTS

The assistance of Matthew Ko in helping develop the simulation environment is acknowledged and greatly appreciated.

REFERENCES

APRIL Robotics Laboratory (2012). APRIL Laboratory : Autonomy * Perception * Robotics * Interfaces * Learning. http://april.eecs.umich.edu.

Brito Palma, L., F. Vieira Coito and P. Sousa Gil (2012). Low order models for human controller - mouse interface. In: *2012 IEEE 16th International Conference on Intelligent Engineering Systems (INES)*. pp. 515–520.

Brown, Michael, Aidan Kehoe, Jurek Kirakowski and Ian Pitt (2010). Beyond the gamepad: HCI and game controller design and evaluation. In: *Evaluating User Experience in Games* (Regina Bernhaupt, Ed.). pp. 209–219. Human-Computer Interaction Series. Springer London.

Chen, J.Y.C., E.C. Haas and M.J. Barnes (2007). Human performance issues and user interface design for teleoperated robots. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* **37**(6), 1231–1245.

Corde Lane, J., C.R. Carignan, B.R. Sullivan, D.L. Akin, T. Hunt and R. Cohen (2002). Effects of time delay on telerobotic control of neutral buoyancy vehicles. In: *IEEE International Conference on Robotics and Automation, 2002. Proceedings. ICRA '02*. Vol. 3. pp. 2874 –2879.

Davis, J., C. Smyth and K. McDowell (2010). The effects of time lag on driving performance and a possible mitigation. *IEEE Transactions on Robotics* **26**(3), 590–593.

Delice, I.I. and S. Ertugrul (2007). Intelligent modeling of human driver: A survey. In: *2007 IEEE Intelligent Vehicles Symposium*. pp. 648–651.

Huang, Albert, Edwin Olson and David Moore (2010). LCM: Lightweight communications and marshalling. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Luck, Jason P., Patricia L. McDermott, Laurel Allender and Deborah C. Russell (2006). An investigation of real world control of robotic assets under communication latency. In: *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. HRI '06. ACM. New York, NY, USA. p. 202209.

MacAdam, Charles C. (2003). Understanding and modeling the human driver. *Vehicle System Dynamics* **40**(1-3), 101–134.

Nise, Norman S. (2004). *Control Systems Engineering*. fourth ed.. John Wiley & Sons.

Sheridan, T. B and W. R Ferrell (1963). Remote manipulative control with transmission delay. *IEEE Transactions on Human Factors in Electronics* **HFE-4**(1), 25– 29.

Toffin, D., G. Reymond, A. Kemeny and J. Droulez (2007). Role of steering wheel feedback on driver performance: driving simulator and modeling analysis. *Vehicle System Dynamics* **45**(4), 375–388.

Ungoren, AY and H Peng (2005). An adaptive lateral preview driver model. *Vehicle System Dynamics* **43**(4), 245–259.

Vozar, Steve (2013). A Framework for Improving the Speed and Performance of Teleoperated Mobile Manipulators. PhD thesis. University of Michigan.