EVALUATING THE FIDELITY OF A SCALE MODEL HEXAPOD MOTION SIMULATOR

Christopher Lawrence, Schmidt Peterson Motorsports; Andrew Borme, Indiana University Purdue University Indianapolis;
Pete Hylton, Indiana University Purdue University Indianapolis

Abstract

Engineers have always endeavored to produce mathematical models of the systems on which they work. The advantages to having accurate mathematical models that represent real-life systems are numerous, and include reduced development time/cost, furthering system optimization, a deeper understanding of the system, predicting behavior in untestable circumstances, and understanding key performance indicators. In a similar vein, simulators attempt to replicate real-life events or experiences, based on underlying mathematical models that drive physical and/or visual stimuli. A key point is that simulators rely partly on human perception to determine the accuracy of the simulation experience. The use of a more complex simulator, as opposed to purely mathematical models, has some advantages specifically for use in areas where humans are integral to the overall functionality of the product. Prevalent examples of this are in the development of high-performance human-operated machines, such as airplanes and automobiles. In these cases, the performance of the machine is intertwined with the human, and the quality of the human-machine interface can be important. In particular, when the engineer is seeking maximum performance, such as in the case of jet fighter aircraft or racing cars, the use of a simulator can allow for performance gains and a deeper understanding of how the human and machine are interacting. In addition, simulators can provide benefits to the human in the form of low-risk training and observation.

A challenge with vehicle simulators that includes the driver is understanding how the platform stimuli correlate with the real stimuli experienced by the driver in racing conditions. If the platform motion stimuli are incorrect or unrealistic, the driver will either adapt to driving the vehicle based on false inputs, or be unable to interpret what the vehicle is doing, and can even lead to the driver becoming physically ill. Thus, it is important to understand how a human perceives the motion of the simulator. This study was supported by the driving simulator developed at Dallara Automobili. Dallara is an Italian company that designs and produces racing chassis for a variety of racing series, notably IndyCar, Indy Lights, and Formula 3. They also have a pair of identical driving simulators, one located in Varano de Melegari, Italy, and one located in Speedway, Indiana. IN this paper, the authors discuss the design, analysis, testing, and evaluation of a scale model of a hexapod motion platform. The platform was instrumented such that the platform's motions could be measured and compared to the lap data provided from the Dallara simulator. These motions were evaluated with and without the use of human perception models that describe how humans perceive motion in quantitative terms.

Introduction

Lap time simulation offers a significant advantage to the motorsports industry. Simulation allows for estimation of vehicle performance around a circuit, which is of critical importance, since that is the singular measure of performance. Nearly unlimited optimization is available in a variety of areas, with very little expense or risk [1]. Simulators also offer the advantage of quick results when compared to a track test. Pais et al. [2] and Mulder noted that, "The advent of car simulators has improved time and cost-effectiveness, while allowing better control and repeatability of the experimental conditions. Furthermore, simulators offer a myriad of possible scenarios while guaranteeing the driver's safety."

A simulator offers the ability to drastically increase track testing time, which can be restricted by budget or rule limitations. A simulator offers a much lower risk to the driver, and requires a smaller crew. It may negate travel costs entirely, in addition to eliminating wear and tear on the car, and running costs such as fuel and tires. It can allow for more risky car setups, or the experimentation with virtual models of components that do not yet exist. It allows for near instantaneous car setup or track changes. Also, Toso and Moroni [3] found that, "Professional driving simulators can be successfully exploited to shorten the traditional design-prototype testing-production process relative to a new racecar." This is beneficial to reducing lead time, development costs, and allowing for preliminary evaluation before car construction begins. In particular, fundamental parameters can be easily changed during simulation.

However, simulators are not perfect. A simulator will always be an approximation of reality. The model inputs to the simulator determine the accuracy, allowing the possibility of false conclusions. For a useful simulation, a wealth of data about the vehicle must be known. Even with a very capable motion platform, the motion and information fed to the driver will not be as good as the physical car and, in some cases, can be misleading or even make the driver sick. Some drivers adapt to the simulator experience better than others, whereas some adapt to driving based on unrealistic stimuli. A typical racing team utilizing a simulator must have a depth of knowledge of the vehicle being simulated and a realistic expectation of the quality results. However, they may not understand how the simulator dynamics and cueing algorithm are stimulating the driver, and this may directly affect the results. By instrumenting the motion platform and comparing the accelerations produced by the simulated vehicle model, the limitations become apparent. If the human perception models are also integrated, driver feedback can be aligned with platform and cueing algorithm strengths and weaknesses. This knowledge allows for a better expectation of the quality of the simulator results, and reduces the chances of false conclusions.

Including the driver in simulations has significant potential advantages, but also significant challenges. The potential advantages consist of increasing simulator accuracy, identification of problems, and understanding car stability issues. Also, by including the driver, subjective feedback from the driver can be gathered, the driver may improve his/her skill, and the performance of the combined driver and car can be ascertained. Some challenges to including the driver are: 1) complicated physical hardware is required; 2) it is significantly more expensive to produce a realistic simulation; and, 3) it is necessary to have a motion platform and cueing algorithm that creates a good illusion of self-motion. When a human driver is included in the simulation, driver inputs are measured. These inputs often do not result in the vehicle being at the limit behavior, so there is no optimization involved here to find the fastest lap time, or best braking point, as the driver is choosing those. This also demonstrates the need for a professional driver when motorsport simulations are being performed. As in real life, the driver must be able to extract performance out of the vehicle to provide value to the simulation effort. In simulation, a single or many real components may be tested. In this case, one real component being tested is the driver. Rather unique to driving simulators, the vehicle mathematical model is also a test subject. In another light, the simulator itself, as a collection of motion stimuli, visual stimuli, and hardware, is a test subject. As mentioned before, the driver makes decisions based on the quality and quantity of visual and motion inputs provided. The interaction between the driver and simulator is of critical importance [4].

In the development and use of driving simulators, there will always be error between the stimuli of the actual vehicle and the stimuli of the platform. If the stimuli are incorrect, the simulator is not perceived as an exact copy of reality to the driver, and can allow for false conclusions. To understand this error, a scale model of a hexapod simulator was designed, built, and tested in this current study to compare the driver stimuli. The platform stimuli were limited by many factors, which included the platform motion envelope, actuator speed and force, and the cueing algorithm. The particular stimuli that were considered were the platform and vehicle motions, specifically the linear accelerations and angular velocities in X, Y, and Z.

Simulation Fidelity

Simulation involves both objective and perceptive fidelity. Objective fidelity is achievable in terms of the mathematical model of the vehicle. Fields such as suspension kinematics, aerodynamics, tires, and vehicle dynamics are well researched, and the car can be modeled within a computer with relative ease [5-9]. Other objective fidelity requirements are an accurate track model. A track laser scan is preferable, if not required, to give correct road profile and roughness [3]. A challenge for creating objective fidelity is replicating the motions of the vehicle utilizing a motion platform. The rapid onset of lateral and longitudinal acceleration, and the rapid change in acceleration directions during successive maneuvers may present a problem. There exists a major compromise in motion platform selection. Smaller, lighter platforms have better response, but larger platforms have higher sustained acceleration and displacement [2]. In any case, it is somewhat unrealistic for a simulator to exactly replicate all accelerations of the vehicle. This necessitates a motion cueing algorithm to determine what motions are desired and possible to replicate.

Perceptual fidelity is achievable in terms of the visual, auditory, and physical vehicle controls. There are advanced graphics packages available, which support driving simulation. Depending on the simulator, the physical user interface can be as accurate as needed. The Dallara simulator features an actual carbon tub within the simulator, including the correct steering wheel, display system, and various controls, such as anti-roll and brake bias adjustments. Automotive and flight simulators also often have an actual cockpit or interior of the vehicle. The sound generation quality does not have a large impact on perceptual fidelity, though its absence is not suggested. A challenge for creating perceptual fidelity is to create an accurate illusion of self-motion. The inability to create a consistent, convincing illusion of self-motion is the primary factor for simulator sickness, where the driver may become disoriented or physically ill. In a study by Pais et al. [2], drivers preferred a motion cueing algorithm that produced no false cues, followed by no motion cueing, followed by a motion cueing algorithm that occasionally had false cues.

Human Perception Transfer Functions

The methods of perception considered in this study were the linear accelerations and angular velocities. The physical human body organs that sense these parameters are the otoliths and the semi-circular canals, respectively. Their transfer functions and motion thresholds were examined by Telban and Cardullo [10]. The transfer functions of the otolith organs are needed to create a transfer function between actual lateral/longitudinal acceleration and sensed lateral/longitudinal acceleration. The transfer functions and perception thresholds were found using Equations (1) and (2):

$$\frac{f_{sensed}}{f_{actual}} = \frac{G_{oto}(K_{oto})(s + A_0)}{(s + B_0)(s + B_1)} \tag{1}$$

$$K_{oto} = K\tau_1\tau_2/\tau_L \tag{2}$$

where,

G_{oto} = 0.0625 m/sec^2 for X motion (longitudinal)
G_{oto} = 0.0569 m/sec^2 for Y motion (lateral)
K = 0.4
τ_1 = 5
τ_2 = 0.016
τ_L = 10
A_0 = $1/\tau_L$
B_0 = $1/\tau_1$
B_1 = $1/\tau_2$

The transfer functions for the semi-circular canals were needed to create a transfer function between the actual yaw, pitch, and roll and the sensed yaw, pitch and roll. The transfer function used by Telban and Cardullo [10] is given by Equation (3):

$$\frac{\omega_{sensed}}{\omega_{actual}} = \frac{5.73(80)s}{(1 + 80s)(1.573s)} \tag{3}$$

The motion thresholds presented by Telban and Cardullo were roll = 3.0 deg/sec, pitch = 3.6 deg/sec, and yaw = 2.6 deg/sec.

Methodology

To fully understand objective and perceptual fidelity, a motion platform with configurable control systems, cueing algorithm, and measurement system was needed. Figure 1 shows a desktop-sized motion platform that was modeled in MATLAB and designed in SOLIDWORKS. Figure 2 shows the manufacture and testing of the platform.

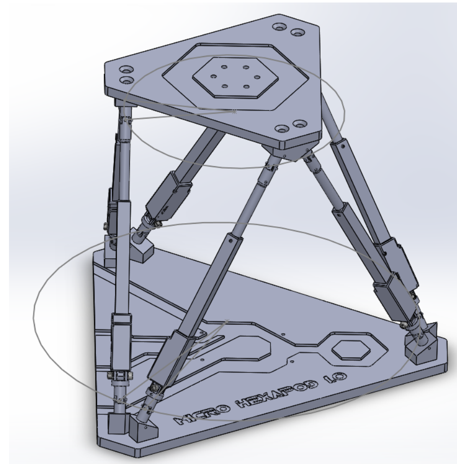

Figure 1. Model of the Mini-Simulator

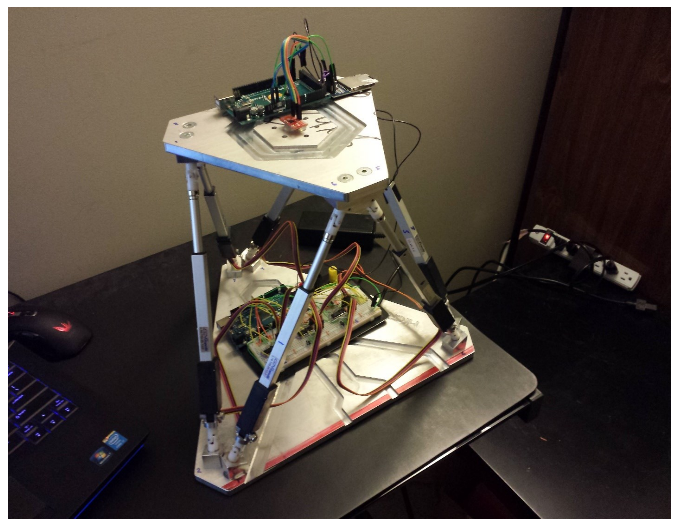

Figure 2. Constructed Mini-Simulator

The motion platform was selected to be a Stewart platform design. A Stewart platform, also known as a hexapod, is a motion platform consisting of two plates connected by six legs with adjustable lengths (actuators). One of the plates is the base, the other is the motion platform. This results in a platform with six degrees of freedom (DOF): 3 DOF in translation and 3 DOF in rotation. Stewart platforms were originally designed to be motion platforms for aircraft simulators, but are now used for driving simulators, machine tool applications, pick-and-place robots, and other applications. For the application of a driving simulator, total motion envelope and maximum platform accelerations are prioritized over platform stiffness and platform positional accuracy [11]. Stewart platforms are classified as parallel robots, where the platform motion in all six DOF is determined by all six actuators in all cases. For example, all six actuators need to move to generate a yaw angle. The benefit of a parallel robot is that the inverse kinematics are relatively simple. The downside is that the motion envelope is highly interconnected between the DOFs. Linear displacement in one direction limits possible roll, pitch, yaw, and linear displacement in the other two directions [12-14]. The Dallara simulator uses a Stewart platform concept, as does the mini-simulator developed for this current project.

This hexapod configuration was selected due to its freedom for selecting motion cueing algorithms, simplistic design, simple inverse kinematics, and common use in the simulator industry. The scale of the platform was limited by actuator availability and budget. The linear actuators selected had the following specifications:

Stroke: 100 mm
Maximum Speed: 32 mm/s
Measurement method: 18 kΩ linear potentiometer
Positional accuracy: 0.4 mm
Backlash: 0.2 mm

Additional mechanical components selected were nylon U joints. Also, 6061 Aluminum was chosen for construction of the base and top plates of the platform, due to its low cost and easy machinability. Before a detailed CAD design was started, the kinematics of the platform were studied using MATLAB Simulink, specifically with SimScape SimMechanics. With the given actuator constraints of minimum and maximum lengths, and the additional length of the universal joints considered, the platform top and bottom radiuses were chosen to balance high platform linear travel in X and Y with high angular displacements in X and Y. This was to allow for the possibility of evaluating tilt coordination, where high angular displacements in X and Y (pitch and roll) were needed when compared to a more typical algorithm, where high linear displacements in X and Y (longitudinal and lateral acceleration) would be needed. The design was loosely based upon a scaled version of the Dallara driving simulator.

The platform also needed an adequate control system to produce the desired motions in the platform. The control system chosen was a standard closed-loop PID controller, implemented using an Arduino Mega 2560 and a series of Texas Instruments SN754410 H-bridge motor drivers. Initially, the PID control was implemented using the external driving capability of MATLAB Simulink. This allowed the model to be created and deployed to the Arduino Mega 2560 without the need to directly interact with the code. This allowed for easy troubleshooting, tuning, and model updates, as it allowed for live updates of the input signals and control system gains. PID gains were tuned by comparing responses to step and sine inputs, and resulted in the following controller:

- Proportional gain: 5
- Integral gain: 0.1
- Derivative gain: 0

Marginally faster responses were noted at higher proportional gain values, but the lower gain value of five was chosen to minimize platform oscillation with a constant signal (no motion). After the errors were worked out and the control system tuned, MATLAB Simulink would not allow for parameters to be directly input using the USB serial connection or specify that certain variables to be stored anywhere other than RAM. This prevented anything other than simple or periodic functions to be implemented on the platform, due to program size that would exceed the storage capability of the Arduino.

The cueing algorithm was chosen to be a hybrid of several of the cueing algorithms commonly used in simulators. Tilt coordination was used for X and Y (longitudinal and lateral acceleration). Due to the lack of Z acceleration data, there was no cueing input for the Z direction. For yaw, the BSS cueing algorithm was implemented; the pitch and roll angles of the car were added to the pitch and roll cues that were created from the tilt coordination algorithm. These algorithms were chosen for their demonstrated effectiveness and simplicity from the studies previously discussed. The gains for the tilt coordination algorithm were varied in order to demonstrate the effect of the cueing algorithm on the motion of the platform. Additionally, the platform inverse kinematics were implemented [11, 14]. The leg lengths were then converted into a 10-bit format for use in the Arduino PID control, where a value of 0 indicated minimum leg length, and a value of 1023 indicated maximum leg length (100 mm).

The platform dynamics were measured to allow for evaluation of the platform motion. The motions of particular interest were the platform accelerations in the X, Y, and Z directions, as well as the yaw, pitch, and roll rates. Because the input data were limited to accelerations in X and Y, as well as yaw rate, those channels in particular were the most important data to acquire for comparison purposes. To measure the platform motion, an integrated 3-axis accelerometer and a 3-axis gyro were selected. The data from these sensors were logged by another Arduino Mega 2560 and stored on an external 16 GB micro-SD card, which was attached to the Arduino. The accelerometer and gyro combination interface used a digital serial I2C interface, and the SD card reader interfaces used a digital serial SPI interface. The measurements were calibrated and converted into m/s^2 and deg/s, respectively, and then written on the SD card in CSV format for later offload. The system logged all six of the parameters and wrote to the SD card at 100 Hz.

There were several unexpected limitations or weaknesses that developed during the course of the study.

- No Z (vertical) acceleration data were available.
- Actuator speed limited the platform's linear and angular acceleration.
- Arduino memory limited the cueing algorithm from running at 50 Hz for full a lap of data.
- Platform measurement gyro used gains from a data sheet, though offsets were measured and accounted for.

Methodology

The motion platform was tested using a simulated racecar data set from the Dallara simulator. This data set was a lap of the Indianapolis Motor Speedway Road Course, and driven by an experienced IndyCar driver using a mathematical model of the 2015 Indy Lights racecar. The data set included all of the necessary parameters to implement the cueing algorithms studied here, including vehicle accelerations, chassis yaw, pitch, and roll, as well as body side slip angle and yaw rate. Note that due to limitations of the actuators, accelerations were scaled. The human perception models show that humans are most sensitive to accelerations and angular velocity and acceleration. Thus, the motion platform accelerations and angular velocities and accelerations were directly compared to the mathematical vehicle model accelerations and angular velocities and accelerations. Additionally, the perceptual fidelity of the motion platform was studied using the transfer functions presented by Telban and Cardullo [10]. The accelerations and angular velocities were compared, as sensed by the human perception models. For example, if the physical racecar on the track produced a motion that the driver cannot sense, then it was not a problem if the platform did not reproduce this motion, despite the obvious discrepancy in objective fidelity.

The tilt translation cueing algorithm included simulator pitch and roll, based on lateral and longitudinal G demands, summed with raw pitch and roll values, based on a one-to-one representation of chassis pitch and roll. The X and Y translation cues were driven by lateral acceleration of the vehicle, and limited by the maximum displacement of the platform. The gains used were:

- Pitch gain: 1/100
- Roll gain: 1/100
- Lateral gain: 1/750
- Longitudinal gain: 1/750

Figure 3 shows a comparison of the raw data. The red lines indicate the motion perception thresholds. That is, any values recorded between the red lines would not be sensed by the average human. An interesting feature of the results is how well the acceleration traces matched, with only a few exceptions. From the lateral acceleration graph of Figure 4, there were several places where the motion platform lagged behind the actual car in generating lateral acceleration. This was due to the limitations of the platform, where at these points one or more of the actuators were at maximum speed. This would be a good issue to fix by increasing actuator speed, however the platform lag was not causing any false cues because the accelerations were still in the correct directions. Another possible way to correct this error would be to reduce the overall magnitude of motion, so that maximum platform acceleration would be less.

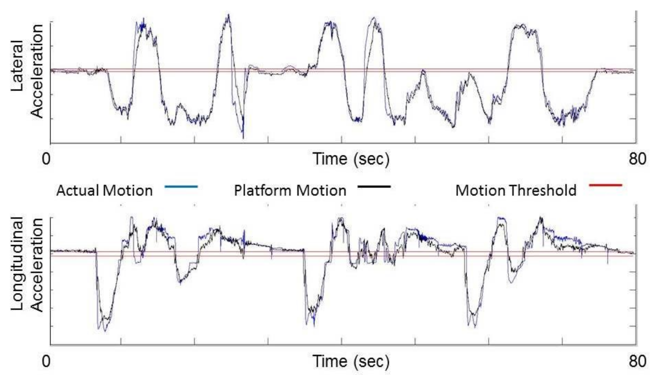

Figure 3. Lateral and Longitudinal Acceleration (m/sec^2) versus Time (sec)
[Note: Acceleration amplitudes are restricted data and were omitted.]

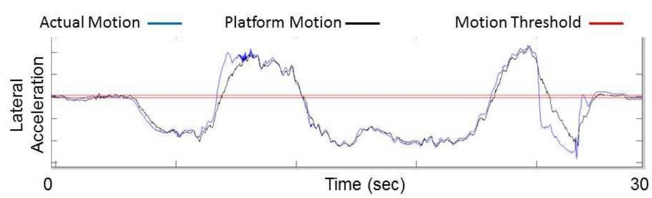

Figure 4. Detail View of Lateral Acceleration (m/sec^2) Error versus Time (sec)
[Note: Acceleration amplitudes are restricted data and were omitted.]

Figure 5 shows the comparison of platform yaw rate with actual vehicle yaw rate, and body side slip rate (BSS rate) versus actual. The cueing algorithm was chosen to follow the BSS angle precisely, thus the BSS rate and the yaw rate of the platform should be close. The platform cannot achieve the yaw rates of the vehicle due to platform limitations, but it can be seen that any change in yaw rate slope (yaw acceleration) was mimicked using the BSS algorithm. Physically, the BSS angle was the slip angle of the chassis of the car, or the angle between where the chassis was pointing and where the chassis was heading. A spike in BSS rate indicates a spin or massive understeer, depending on the sign and the direction of the corner. There were several spots where the platform yaw rate was opposing the actual BSS rate; Figure 6 represents this as a false cue.

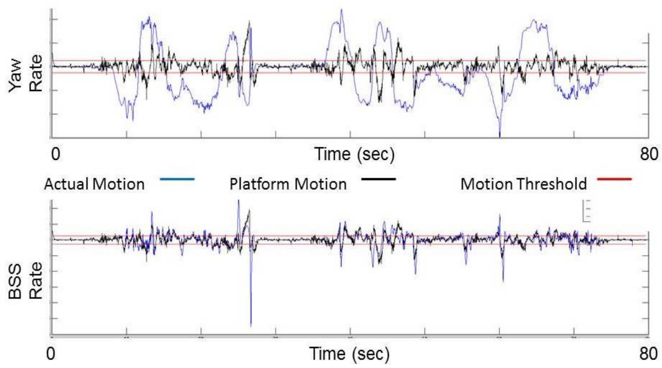

Figure 5. Yaw and BSS Velocity (deg/sec) versus Time (sec) [Note: Amplitudes are restricted data and were omitted.]

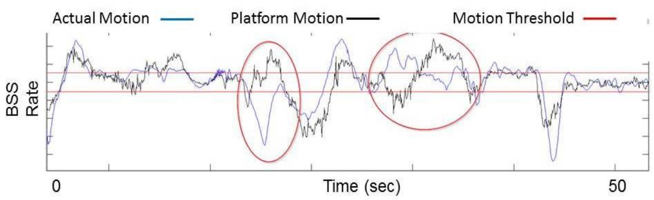

Figure 6. BSS Rate (deg/sec) versus Time (sec). Detailed View of False Cues
[Note: Amplitudes are restricted data and were omitted.]

It can be observed that nearly all of the pitch and roll inputs from the actual car were not perceived by the average human (i.e., they fall between the red lines representing the threshold). However, the pitch and roll rates of the platform were much higher, due to the tilt coordination algorithm rolling and pitching the platform to create the lateral and longitudinal acceleration cues. Clearly, this algorithm did not pitch or roll the platform slow enough to stay under the perception thresholds of the driver. This showed the basic compromise in the tilt coordination algorithm. The pitch and roll rates can be reduced to eliminate the false pitching and rolling cues, but then the magnitude of the acceleration cues would decrease and have delayed onset.

The graphs in Figures 7 and 8 add the human perception transfer functions discussed thus far to show how well humans perceive the motion of the racing car and platform, respectively. The figure was a direct comparison of both the lateral acceleration and yaw rate of the platform, before and after the human perception transfer functions, to illustrate the differences between reality and perception. It can be seen that the transfer functions do not show drastic differences between the actual and perceived motions. The lateral acceleration motions were mostly perceived to be the same, with a reduction in magnitude for lower acceleration values, and an increase in magnitude for high changes in acceleration (high jerk). This might indicate that an algorithm for lateral and longitudinal motion based on matching jerk of the platform and vehicle may produce good simulator results. The yaw rate compared to the perceived yaw rate showed attenuation of the signal for high yaw rates. It also showed a high response to yaw acceleration, but a decreasing response to sustained constant yaw rate. This also indicated that an algorithm based on yaw acceleration might produce good simulator results. It also showed why the BSS algorithm had good success, due to BSS angle representing the changes in yaw angle. Based on the perceived yaw rate, the platform may be able to achieve a one-to-one algorithm.

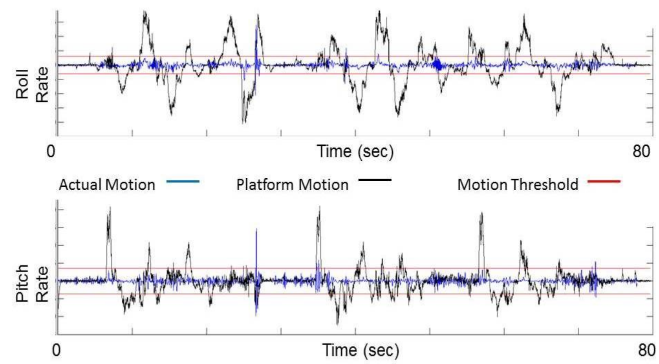

Figure 7. Pitch and Roll Velocity (deg/sec) versus Time (sec) with Human Perception Transfer Functions
[Amplitudes are restricted data and were omitted.]

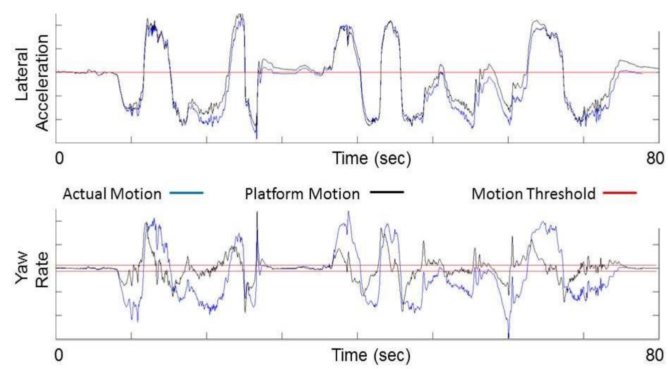

Figure 8. Lateral Acceleration (m/sec^2) and Yaw Rate (deg/sec) versus Time (sec). Comparison between Measured and Perceived Values from the Actual Vehicle
[Amplitudes are restricted data and were omitted.]

The graphs of Figures 9-11 echo Figures 3, 5, and 7, where accelerations and angular velocities from the vehicle and simulator were compared, but with all of the data processed using the given human perception transfer functions to illustrate the human perception of the motion.

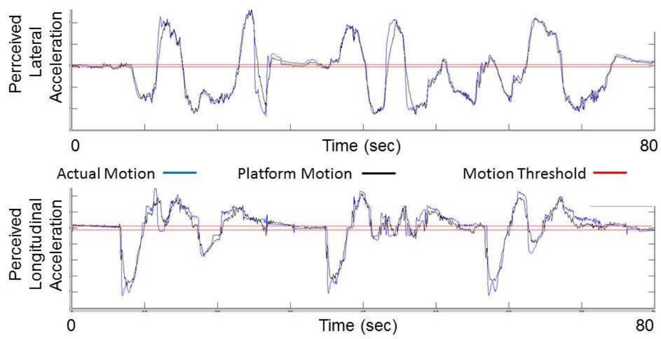

Figure 9. Sensed Lateral and Longitudinal Acceleration (m/sec^2) versus Time (sec)
[Amplitudes are restricted data and were omitted.]

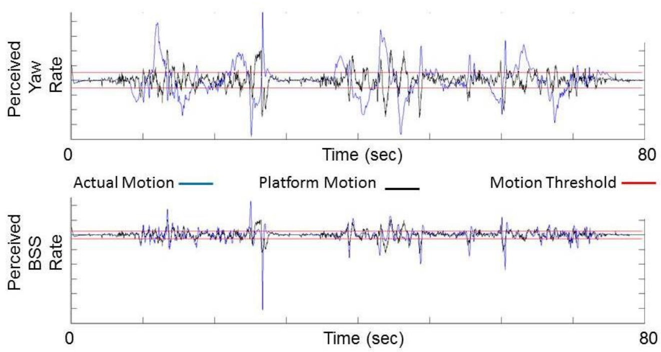

Figure10. Sensed Yaw and BSS Velocity (deg/sec) versus Time (sec)
[Amplitudes are restricted data and were omitted.]

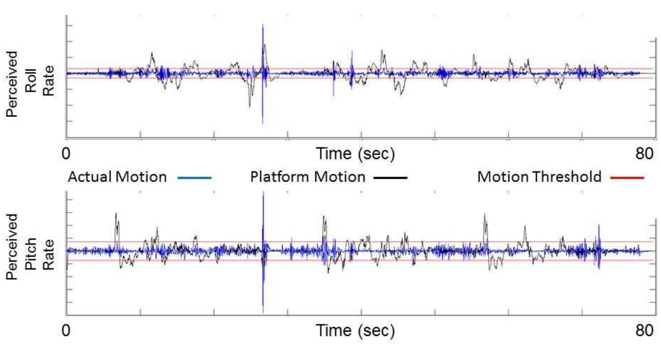

Figure 11. Sensed Pitch and Roll Velocity (deg/sec) versus Time (sec)
[Amplitudes are restricted data and were omitted.]

The perceived acceleration traces were very similar to the actual acceleration traces, and the conclusions about the algorithm effectives were similar. There were still a few areas on the lateral acceleration trace where the platform speed limitation impacted the platform acceleration preventing better cueing. The reduction in sensitivity to yaw has benefits in reducing the perceived magnitude of the false cues noted above. Figure 12 shows that there was only one major false cue and one minor false cue observed in the BSS rate graph. Overall, more of the cues were below the perception threshold, and could be filtered out. The perception modeling had the biggest impact on the large false pitch and roll cues produced by the simulator, and it can be seen how much less magnitude of the pitch and roll motions were perceived. However, there were still 18 false perceived roll cues and 18 false perceived pitch cues.

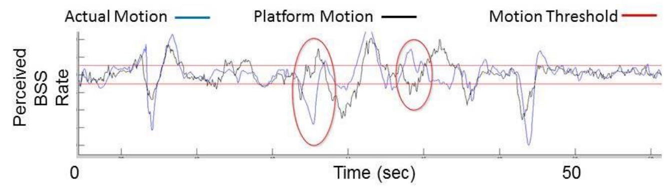

Figure 12. BSS Rate (deg/s) versus Time (s). Detailed View of False Cues
[Amplitudes are restricted data and were omitted.]

Conclusions

A scaled down model of a hexapod driving simulator was designed, built, tested, and evaluated using lap data. The platform kinematics were studied, actuators and electronics selected, key dimensions chosen, and the platform was constructed. A tilt coordination cueing algorithm was created in MATLAB and implemented using Arduino code. The platform linear accelerations and angular velocities were measured. Based on the platform accelerations and angular velocities, the cueing algorithm demonstrated very good acceleration results and moderately good yaw results. The pitch and roll results showed significant false cues, which may be a detriment to driver performance and realism in the simulator. When these results were viewed after being transformed by the human perception models, the acceleration results were very good. The false cues in the yaw direction were minimized, but still existed. To a much higher degree, the pitch and roll false cues were minimized, but there were still pitch and roll motions above the driver perception threshold that were not present in the lap data. This cueing algorithm could be improved by reducing the gain for the tilting of the platform and increasing the gain for the translation of the platform. This would reduce the pitch and roll false cues, at the expense of reducing the linear acceleration magnitudes.

The platform developed in this study was limited by the maximum speed of the actuators used. If faster actuators could be developed or purchased, a more translation-biased cueing algorithm could produce high linear accelerations based on platform motion alone. Additionally, based on the human perception results from this simulation, a translation based algorithm that correlates vehicle jerk with platform jerk may produce good results, especially in the case of a fast platform with a limited-motion envelope. Also, based on the perceived yaw, an algorithm that attempts to do a one-to-one perceived yaw seems possible, and would provide the driver with more information about car balance, which is a traditional simulator shortcoming.

Acknowledgements

The research conducted in this study was funded by a two-year grant from the Indiana Economic Development Corporation (IEDC), in partnership with the Purdue School of Engineering and Technology at Indiana University-Purdue University, Indianapolis (IUPUI) and Dallara. Without the support of the IEDC and Dallara's collaborative efforts with IUPUI, this project would not have been possible.

References

[1] Siegler, B., Deakin, A., and Crolla, D. (2000). Lap Time Simulation: Comparison of Steady State, Quasi - Static and Transient Racing Car Cornering Strategies. *SAE Technical Paper 2000-01-3563*, doi:10.4271/2000-01-3563.

[2] Pais, A., Wentink, M., Paassen, M., & Mulder, M. (2009). Comparison of Three Motion Cueing Algorithms for Curve Driving in an Urban Environment. *Presence: Teleoperators and Virtual Environments*, 18(3), 200-221.

[3] Toso, A., & Moroni, A. (2014). Professional Driving Simulator to Design First-Time-Right Racecars. *SAE Technical Paper 2014-01-0099*, doi:10.4271/2014-01-0099.

[4] Waeltermann, P., Michalsky, T., and Held, J. (2004). Hardware-in-the-Loop Testing in Racing Applications. *SAE Technical Paper 2004-01-3502*, doi:10.4271/2004-01-3502.

[5] Milliken, W., & Milliken, D. (1995). *Racecar Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers

[6] Pacejka, H. (2012). *Tire and Vehicle Dynamics (3rd ed.).* Waltham, MA: Butterworth-Heinemann.

[7] Gillespie, T. (1992). *Fundamentals of Vehicle Dynamics.* Warrendale, PA: Society of Automotive Engineers.

[8] Guiggiani, M. (2014). *The Science of Vehicle Dynamics: Handling, Braking, and Ride of Road and Racecars.* New York, NY: Springer.

[9] Sergers, J. (2014). *Analysis Techniques for Racecar Data Acquisition (2nd ed.).* Warrendale, PA: Society of Automotive Engineers.

[10] Telban, R., & Cardullo, F. (2005). Motion Cueing Algorithm Development: Human-Centered Linear and Nonlinear Approaches. *NASA Technical Paper NASA/CR-2005-213747*.

[11] Gong, Y. (1992). *Design Analysis of a Stewart Platform for Vehicle Emulator Systems*. (Master's thesis). Retrieved from http://dspace.mit.edu/bitstream/handle/1721.1/34314/26408968-MIT.pdf?sequence=2

[12] Garrett, N., & Best, M. (2013). Model Predictive Driving Simulator Motion Cueing Algorithm with Actuator-Based Constraints. *International Journal of Vehicle Mechanics and Mobility,* 51(8), 1151-1172.

[13] Korobeynikov, A., & Turlapov, V. (2005). Modeling and Evaluating of the Stewart Platform. *Proceedings of the International Conference Graphicon 2005,* doi:10.1.1.189.5199.

[14] Jakobovic, D., & Jelenkovic, L. (2002). *The Forward and Inverse Kinematic Problems for Stewart Parallel Mechanisms.* Retrieved from http://alvarestech.com/temp/RoboAseaIRB6S2-Fiat/CinematicaHexapode.pdf

Biographies

CHRIS LAWRENCE is a simulation engineer for Schmidt Peterson Motorsports. He received his bachelor's degree in Mechanical Engineering from Colorado State University and his master's degree from Indiana University Purdue University Indianapolis. Mr. Lawrence may be reached at chrisl@spmindycar.com

ANDREW BORME is a Senior Lecturer of Motorsports Engineering at Indiana University Purdue University Indianapolis. He received his BS in engineering from Rensselaer Polytechnic Institute and MS in engineering from California State University Long Beach. Mr. Borme may be reached at aborme@iupui.edu

PETE HYLTON was a Professor of Motorsports Engineering at Indiana University Purdue University Indianapolis. He earned his BS degree in engineering from Rose-Hulman Institute, MS degrees in engineering and math from Purdue University, and an EdD from Grand Canyon University. Dr. Hylton may be reached at phylton@iupui.edu