A Simulation-Based Control Interface Layer for a High-Fidelity Anthropomorphic Training Simulator

Kyle Nelson, Timothy Black, Doug Creighton, Saeid Nahavandi
Centre for Intelligent Systems Research, Deakin University
Geelong, Victoria, Australia
kjnel, timothy.black, doug.creighton, saeid.nahavandi@deakin.edu.au

ABSTRACT

Deakin University's futuristic Universal Motion Simulator will overcome the limitations of current motion simulator platforms by employing an anthropomorphic robot arm to provide the motion fidelity necessary to exploit the potential of modern simulation environments. Full motion simulators frequently utilize Stewart platforms to mimic the movement of vehicles during simulation. However, due to the limited motion range and dexterity of such systems, and their inability to convey realistic accelerations, they are unable to represent accurate motion characteristics. The Universal Motion Simulator aims to close the gap between the limitations of the current motion technology and the real world, by introducing a flexible, modular, high-fidelity motion system that can be used for a variety of immersive training applications. The modular nature of the design allows interchangeable and configurable simulation pods to be attached to the end effector.

Existing robot-based motion platforms use a direct control input via hardware to control the motion of the robot. In this paper we present a methodology that increases the capability of this novel motion simulator by introducing a simulation-control interface layer. The Control Interface Layer allows the user to interact with the simulation environment, unlike current direct hardware control interfaces on this type of motion platform. This layer enables anthropomorphic motion platforms to become a more cost-effective solution for re-configurable motion simulation hardware.

Results of this research show that a simulation-control layer allows the novel motion simulator to accurately reproduce the motion of a simulated vehicle while introducing minimal latency within the control loop and greatly increasing the overall functionality of the simulator. This motion platform will open new research opportunities in the study of human-machine interaction, human psychophysics, and the evaluation of human performance in virtual training situations.

ABOUT THE AUTHORS

Kyle Nelson received the B.Eng. (hons.) in Mechatronics and Robotics from Deakin University, Geelong, Australia, in 2009. He is currently pursuing a Ph.D. in vision systems and image processing at the Centre for Intelligent Systems Research, Deakin University, Geelong. His current research interests include robot-based motion simulation, networked vision systems, image processing and image resolution enhancement.

Timothy Black received a B.Eng. (hons.) in the field of Robotics in 2004 and a PhD degree in wireless sensor networks in 2009 from Deakin University, Geelong, Australia. Currently a research fellow within the Centre for Intelligent Systems Research at Deakin University, Dr Black has been actively involved in intelligent robotics, virtual training, robot-based motion simulation, and discrete event simulation research.

Doug Creighton received a B.Eng. (hons.) in Systems Engineering and a B.Sc. in Physics from the Australian National University in 1997, where he attended as a National Undergraduate Scholar. He spent several years as a software consultant prior to obtaining his PhD degree in simulation-based optimization from Deakin University in 2004. He is currently a research academic and stream leader with the Centre for Intelligent Systems Research (CISR) at Deakin University. Dr Creighton has been actively involved in modeling, discrete event simulation, intelligent agent technologies, HMI and visualization and simulation-based optimization and training research.

Saeid Nahavandi received a B.Sc. (hons.), M.Sc., and Ph.D. degrees in automation and control from Durham University, Durham, U.K. He is the Alfred Deakin Professor, Chair of Engineering, and the director for the Center for Intelligent

Systems Research (CISR), Deakin University, Geelong, Vic, Australia. He has published over 300 peer reviewed papers in various international journals and conferences. He designed the world's first 3-D interactive surface/motion controller. His research interests include modeling of complex systems, simulation-based optimization, robotics, haptics and augmented reality. Dr. Nahavandi was a recipient of the Young Engineer of the Year Title in 1996 and six international awards in Engineering. He is the Associate Editor of the IEEE SYSTEMS JOURNAL, an Editorial Consultant Board member for the International Journal of Advanced Robotic Systems, an Editor (South Pacific Region) of the International Journal of Intelligent Automation and Soft Computing. He is a Fellow of Engineers Australia (FIEAust) and IET (FIET).

A Simulation-Based Control Interface Layer for a High-Fidelity Anthropomorphic Training Simulator

Kyle Nelson, Timothy Black, Doug Creighton, Saeid Nahavandi
Centre for Intelligent Systems Research, Deakin University
Geelong, Victoria, Australia
kjnel, timothy.black, doug.creighton, saeid.nahavandi@deakin.edu.au

INTRODUCTION

Full motion simulators have relied on Stewart Platform technology to emulate vehicle motion for several decades. This concept emerged in 1965 (Stewart, 1965) and through a process of continual evolution remains the technology of choice in vehicle simulation today. The success of Stewart Platform-based simulators can be attributed to the high payloads and high momentary acceleration possible with this design. However, roll, pitch and yaw in a Stewart Platform-based motion simulator are considerably limited and rotation is restricted significantly at the limits of the motion envelope, despite exhibiting 6 degrees of motion. As a result of the hexapod design, current full motion simulators may only facilitate linear displacement in the order of one meter. This makes it very difficult to convey realistic accelerations, particularly in the forward direction.

As modern simulation systems continue to improve, advanced high-fidelity simulation environments frequently appear. New software architectures (Zheng et al., 2009, Cremer & Papelis, 1993) have allowed simulation environments to become increasingly complex (Zheng et al., 2008, White & Padfield, 2004). As high-fidelity simulation software continually improves, advanced platform designs are required to exploit the full capability of these systems. The emergence of anthropomorphic robot-based motion simulators (Teufel et al., 2007) provides one feasible solution to this problem. This concept presents a system that closes the gap between the limitations of current Stewart-based platforms and the real world. Now that several robot-based motion simulators have emerged (Kecskeméthy, Masic & Tändl, 2008, Schaetzle, Preusche & Hirzinger, 2009, Teufel et al., 2007) a new challenge lies in interfacing advanced simulation environments with this revolutionary simulator design. In this paper we present a Control Interface Layer that extends the capability of robot-based simulators and links realistic simulation software with the novel simulator hardware.

Figure 1. A simulation model of the Universal Motion Simulator

BACKGROUND

Over the past 40 years several high-fidelity motion simulators have been developed, primarily with the aim of establishing a wider motion envelope. The Large Amplitude Multi-mode Aerospace Research Simulator (LAMARS) was commissioned at the Aerospace Vehicles Technology Assessment and Simulation Laboratory at Wright-Patterson Air Force Base in the United States in 1974 (James & Negaard, 1981). This particular simulator features a spherical aircraft cockpit and display screen at the end of a 30 ft mechanical beam. This allows the simulator 5DOF including roll, pitch, yaw and vertical and lateral movement of the beam. Another well known high-fidelity motion simulator is the Vertical Motion Simulator (VMS), developed at NASA in 1979 (Aponso, Beard & Schroeder, 2009). The VMS consists of a 4DOF cockpit mounted on a 2DOF large-amplitude platform. The VMS cockpit can move approximately 10.5m horizontally and 15m vertically. A more recent development is the Desdemona concept designed by TNO Human Factors in conjunction with AMST Systemtechnik (Wentink et al., 2005). This particular simulator involves a fully gimbaled cockpit capable of unlimited rotation in all directions, with a range of motion of 8m in the horizontal direction and 2m vertically.

While these simulators improve the work envelope available for simulation, they remain exceptionally mechanically complex. As a result they are all extraordinarily expensive one-off systems. Novel technologies to generate the complex range of motion required to exploit the potential of current high-fidelity simulation environments have not yet been developed in a flexible, compact and cost effective motion platform. The following sections discuss a revolutionary anthropomorphic robot-based training simulator that addresses the need for a new generation of high-fidelity motion simulators.

THE UNIVERSAL MOTION SIMULATOR

In the search for a new-age simulator design that can accommodate the high-fidelity motion demanded by modern simulation environments recent research has focused on robot-based simulation platforms (Beykirch et al., 2007, Kecskeméthy, Masic & Tändl, 2008, Niccolini et al., 2009, Nusseck et al., 2008, Schaetzle, Preusche & Hirzinger, 2009, Teufel et al., 2007). One such concept is the haptically enabled Universal Motion Simulator (UMS) being developed at Deakin University, which employs an anthropomorphic robot that has a wide range of motion in six degrees in freedom. This next generation in simulation systems features a far greater range of motion, flexibility and realism.

Creating a simulator whereby the user is maneuvered by a back-drivable robot enables large-amplitude, dexterous motion not previously possible with a Stewart Platform. A simulator based on a serial robot permits motion remarkably similar to that which we may experience in a real vehicle, such as rolling a motor vehicle and inverted flying or loops in an aircraft. Not only does a simulator of this nature feature a large motion range, but the ability to also combine rotational and translational motion allows the system to behave very similarly to a real vehicle. This achieves a level of realism far beyond that demonstrated by current Stewart Platform simulators. While Stewart Platforms can achieve momentary high acceleration, the larger work envelope of simulators such as the UMS means that high acceleration can be maintained for extended periods. Robot-based motion simulators will enable occupants to experience gravitational forces approaching that felt in a real aircraft or motor vehicle maneuvers and velocities of up to 120° per second.

The use of an anthropomorphic robotic arm as a motion platform permits a design with significantly reduced mechanical complexity compared to other high-fidelity motion simulators and also allows high accuracy and dexterity in movement. Using a robot as the primary structure in a motion simulator also facilitates the design of interchangeable and configurable cockpits, which can be connected to the end effector of the robot. For example, modular pods can be designed to emulate a range of different aircraft, making it possible for a training provider to obtain any number of cockpits and only a single motion simulation platform. Hence, the Universal Motion Simulator design not only features increased capability compared to existing systems but the flexible hardware configuration reduces the cost of ownership, as does the use of a mass produced robotic arm.

Existing designs, such as the MPI Motion Simulator (Teufel et al., 2007), feature various passive controls allowing the user to control the motion simulator. The UMS however, will feature a helicopter cyclic with haptic feedback, allowing the user full immersion in the simulation environment. As shown by Teufel et al. (2007), the design of the control structure in the MPI Motion Simulator is such that the user has direct control of the simulator platform via

hardware such as a cyclic, which in turn results in movement of the robotic arm. For this hardware to be integrated into the motion simulator domain the user must interact with a simulation environment. The following section presents a novel Control Interface Layer for the UMS which enables the introduction of a simulation environment into the control loop.

CONTROL INTERFACE LAYER

In order to create a motion simulation experience whereby the user is fully immersed in the simulation environment, it is critical that the user interacts directly with this environment, rather than the system's motion hardware. The Control Interface Layer (CIL) has been designed to separate the simulation environment from the motion platform, while at the same time providing a control interface between the two (Figure 2).

The control structure of the MPI Motion Simulator (Teufel et al., 2007) shows that in this system the robot reacts to a single user input only. The user commands a desired movement via a cyclic directly linked to the control system, which is converted to joint angles for the robot. In contrast, the Universal Motion Simulator includes a separate simulation control loop in the first stage of the system (Figure 2) which provides visual cues to the user. In the UMS system the user controls a simulated vehicle, rather than the motion of the robot.

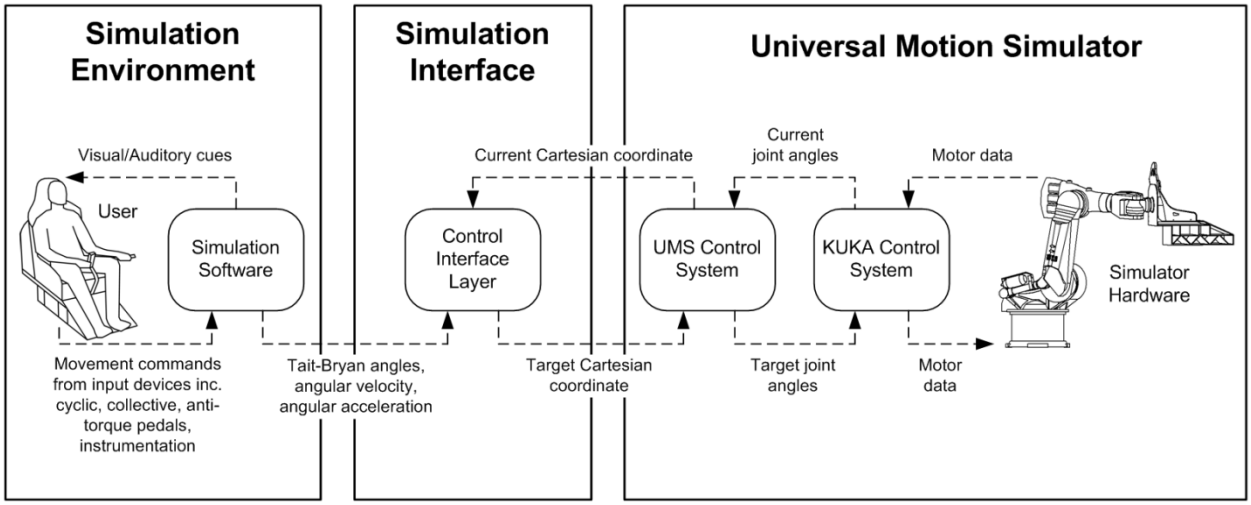

Figure 2. Overview of the Universal Motion Simulator control system

This simulation loop involves the user being prompted by the simulation environment and then reacting to the given conditions, which is in turn interpreted by the simulation software and the cycle continues. Data from the simulation control loop is also supplied to the UMS Control System via the CIL (Figure 2). Hence, the UMS system will react to user input as well as environmental factors from the simulation, as opposed to reacting only to a single input device. This results in the user interacting only with the simulation environment, rather than the robotic hardware, which greatly increases the overall level of realism and enables the user to become immersed in the simulation more easily.

A typical example of the influence environmental factors might have on a simulation could involve a user driving a simulated car around a right-hand bend on a gravel road. In this situation the user would simply steer to the right to negotiate the corner. If a direct control input system was used in this case the only available input would be the commanded movement to the right. However, a simulation environment could account for the bumps in the gravel road, the possible drift as the car rounds the corner and other factors such as the weather, resulting in a more comprehensive representation of the vehicle movement being available as input to the simulator.

Using the CIL to interface with potentially any simulation environment allows the UMS to fully utilize the extremely realistic simulation environments that are emerging. An alternative to extracting motion data from a simulation environment is to design a dynamic model of the vehicle to be simulated. However, in doing so one would most likely need to limit the complexity of the system and thus its realism. As the UMS has been designed as

a re-configurable simulator dynamic models of each vehicle to be simulated would be required, along with a further system of providing visual cues to the user. A simulation environment however serves a dual purpose; facilitating visual cueing and motion control in the UMS system. Simulation environments also allow a wide range of control devices such as joysticks, pedals, dials and gauges to be interfaced with the UMS via simple USB connection, in contrast to the complicated interfaces and measurement devices most likely needed to interface similar devices using a direct control input method. Hence, the utilization of simulation environment input to the UMS is not only realistic but the most effective solution for a re-configurable simulator design.

Simulation Environment-CIL Interaction

In the first module of the UMS control system advanced helicopter simulation software provides a visual stimulus with which the user can interact. Reactions by the user then prompt further changes in the simulation environment, which the UMS must reproduce. In order to achieve this, the CIL retrieves motion data from the simulation environment to be communicated with the simulator hardware.

The CIL has the capability to extract a wide range of information from the simulation environment including vehicle speeds, current G-forces, angular moments, side slip, latitude, longitude and altitude. In its current configuration the CIL accepts Tait-Bryan angles, angular velocity and angular acceleration information, defined as:

$$P_v = [r, p, y] \quad (1)$$

$$\omega_v = [\dot{r}, \dot{p}, \dot{y}] \quad (2)$$

$$\alpha_v = [\ddot{r}, \ddot{p}, \ddot{y}] \quad (3)$$

CIL-UMS Interaction

Upon receiving movement information from the simulation environment the CIL performs complex kinematic mapping functions in order to derive motion commands for the UMS, which is beyond the scope of the paper. This process results in a desired Cartesian coordinate for the Universal Motion Simulator:

$$P_{target} = [x, y, z] \quad (4)$$

The required position, P_{target} is then forwarded to the UMS Control System. There is also capability for the CIL to receive feedback from the UMS Control System regarding achieved coordinates so that during kinematic mapping the CIL can make any required adjustments to the simulator path.

The final stage in the control process includes the UMS Control System and the KUKA Control System, which mimics the system described by Teufel et al. (2007), whereby the UMS Control System converts the desired Cartesian coordinate P_{target} to joint angles for the KUKA Control System, which controls the actuation of the simulator arm.

CIL Software and Evaluation

Apart from acting as a communications interface between the simulation software and the Universal Motion Simulator, the CIL also performs various additional functions including movement safety monitoring and motion cueing. The software created for the CIL also features an extensive GUI that provides the user with a greater understanding and complete control over the system. The GUI allows the user to monitor data being exchanged with the simulation software such as acceleration, velocity and position information, and a communications tab allows the user to alter UDP connection settings. The user can also monitor the coordinate data sent to the UMS Control System and all data can be logged for later analysis of the simulation.

The CIL software has been extensively tested using flight simulation software and a visual simulation environment with a model that accurately represents the operation of the UMS. This arrangement demonstrates the mapping of motion cue input from a virtual helicopter to the motion of the simulator model. The result is a system whereby the user can fly a helicopter in the simulation software and the CIL allows the visual simulator model to follow the motion of the aircraft in real time (Figure 3).

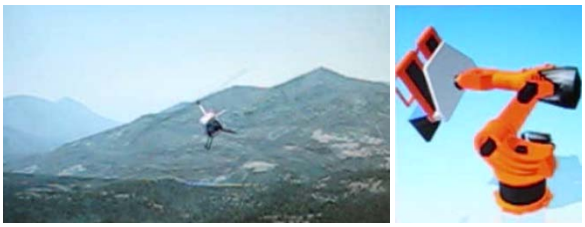

Figure 3. Real-time kinematic mapping between simulation software and a visual model of the UMS

Movement Safety

One of the major concerns relating to robot-based motion simulation is passenger safety. The workspace for the motion simulator must be designed such that the simulation pod can never collide with the ground or any part of the anthropomorphic arm. The establishment of the CIL in the UMS control loop introduces an opportunity to implement a range of safety measures. For example, while the UMS and KUKA Control Systems will perform their own safety assessment the CIL presents an additional point to check that commanded orientations lie safely within the workspace of the motion simulator.

As the CIL interfaces between the Universal Motion Simulator hardware or a visual model and any simulation software, a range of testing configurations are made possible. Connecting simulation software to a visual model of the UMS via the CIL allows great flexibility during the design phase without the need for the UMS hardware. This enables researchers to test and refine changes to motion cueing algorithms and safety checks before they are trialed on expensive equipment. Also, as hardware is not required in this configuration, many different researchers can experiment with the UMS and be confident that developed routines will function as expected when experiments are conducted with the UMS hardware.

As the Universal Motion Simulator accepts motion commands from the simulation environment via the CIL, the motion platform can potentially accept input from any instance of the simulation software. It is also possible for the CIL to interface with multiple instances of the software, which could enable dual control in a situation involving a student and instructor, for example. A key benefit of the design is that the input to the UMS has been isolated from the simulator hardware so that a user need not be present to operate the simulator. In this scenario the CIL enables the UMS to be operated remotely from a control room by redirecting the input to the CIL, allowing researchers to carefully observe, measure and refine motion cueing algorithms before experimenting with human pilots.

Interchangeability

As indicated, the UMS is currently configured as a helicopter simulator, using highly realistic flight simulation software that allows instructors to control details from the aircraft design and the flight environment and terrain to the weather conditions and equipment failures via the CIL. The CIL not only enables such simulation environments to be integrated with the UMS, but allows the hardware platform to be quickly transformed into any type of transport simulator. The CIL makes it possible for potentially any vehicle simulation software to be linked to the UMS. Combined with the ability to connect a range of simulation pods to the end effector of the UMS, the motion simulator could quickly become a fighter jet, naval destroyer, submarine, armored personnel carrier or battle tank. As a result, this innovative motion simulator can be used to investigate pilot-vehicle interaction, pilot training or the development or any type of vehicle.

EXPERIMENTAL RESULTS

One of the most important considerations with respect to the development of the CIL is the accuracy with which the Universal Motion Simulator follows the roll, pitch and yaw of the aircraft in the helicopter simulation environment. If the mathematical transformations within the CIL are correct, the UMS should follow exactly the path flown by a helicopter in the simulation software. The other main factor of importance is the speed at which the system operates. If the CIL processes data too slowly it will result in latency in the system and the motion simulator will not accurately follow the user input and visual cues from the simulation software.

To test the capabilities of the CIL a series of four experiments were conducted to record the exact data sent and received at each point in the system. All experiments undertaken involved the use of two separate computers. One computer was used to operate the helicopter simulation software while the CIL and visual model of the UMS were executed from a second computer, as this is how the software is arranged for operation with the Universal Motion Simulator hardware. Due to the extremely high graphics demand of the simulation software this program will always be executed from a separate computer.

The actual data recorded during the experiments included the roll, pitch and yaw of the aircraft from the simulation environment, the corresponding variables received at the CIL as well as the commanded joint angles sent to the visual model and the joint angles received by the visual model. The experiments conducted also aimed to measure the time taken for the CIL to process data and the additional time taken to transfer data to the visual model. Hence, during the experiments timing data was also recorded at all three points in the system.

The CIL recorded the exact time that it received each packet of data from the simulation software as well as the exact time that it sent a packet of data to the visual model, while the model recorded when it received each packet from the CIL. All timing data was recorded to millisecond accuracy as this is the most precision possible using the computer available. The collection of timing data allowed an analysis of not only the accuracy of the system but the exact time at which each application sent or received a data packet.

Table 1. Sample experimental data

Simulation software			**Control Interface Layer**					**Visual model of the UMS**			
Roll (°)	***Pitch (°)***	***Yaw (°)***	***Rec. Time (s.ms)***	***Roll (°)***	***Pitch (°)***	***Yaw (°)***	***Sent Time (s.ms)***	***Rec. Time (s.ms)***	***Roll (°)***	***Pitch (°)***	***Yaw (°)***
-2.83889	-0.21103	113.98201	13.718	-2.83889	-0.21103	113.98201	13.718	13.718	-2.838890	-0.211030	113.982010
-2.8226	-0.20262	113.98282	13.734	-2.8226	-0.20262	113.982819	13.750	13.750	-2.822600	-0.20261$\dot{9}$	113.982819
-2.80577	-0.19422	113.98351	13.765	-2.80577	-0.19422	113.983513	13.765	13.765	-2.80576$\dot{9}$	-0.194220	113.983513
-2.78862	-0.18689	113.98407	13.796	-2.78862	-0.18689	113.98407	13.796	13.796	-2.78861$\dot{9}$	-0.186890	113.984070
-2.77098	-0.18138	113.98449	13.812	-2.77098	-0.18138	113.984489	13.812	13.812	-2.770980	-0.181380	113.984489
-2.75279	-0.1783	113.98481	13.843	-2.75279	-0.1783	113.98481	13.843	13.843	-2.752790	-0.17829$\dot{9}$	113.984810

Table 1 shows an example of the data that was collected from each of the three applications during the experiments conducted. The sample data shown was taken from the first trial in Experiment 1, under autopilot conditions.

To enable an analysis of the consistency of the experimental results, the four separate experiments were conducted multiple times. Experiment 1 involved the helicopter being flown under autopilot conditions. During this experiment the helicopter was first airborne and then the autopilot function was turned on. Once the autopilot took control of the aircraft the cyclic and pedals were not touched for the duration of the trial. Ideally this experiment should mimic realistic variations in roll, pitch and yaw that would occur during a normal flight.

Experiment 2 involved the helicopter being flown in an extreme roll situation. To achieve this, the aircraft was again airborne; the cyclic was then held to the extreme right for the duration of the flight. Experiment 3 required that the aircraft was forced into an intense pitch situation. This was achieved by holding the cyclic directly backwards for the duration of the flight, forcing the helicopter to complete constant backward loops. The final experiment conducted

involved excessive variation of the helicopter yaw. For this experiment the anti-torque pedals were pushed to their extremities, forcing the helicopter in to a constant clockwise spin. During the final three experiments, apart from the variable being tested, no other flight controls were touched during the test flight. Hence, the helicopter was not stabilized and would eventually crash. As a result many trials were conducted of each experiment to ensure an adequate amount of data was recorded.

To enable a comparison of the results from the four experiments, Table 2 highlights key statistics from each. One trial from each experiment was selected for comparison including trial 1 from Experiment 1, trial 7 from Experiment 2, trial 2 from Experiment 3 and finally trial 5 from Experiment 4. As can be seen from Table 2 the results across all trials were very consistent. The first section of the Table shows the duration of the trials selected and the total number of data packets that were exchanged during the trial. From the timing data recorded the achieved UDP rate was calculated. The achieved rate ranged from approximately 41 data packets per second to over 45.5. From the data analyzed it appeared that the UDP rate increased with the duration of the trial and stabilized at around 45 packets per second. The most significant aspect of this result is that the required UDP rate in the simulation software was set to the maximum allowable level. Hence, the measured UDP rate of 45.5 represents the maximum achievable rate given the experimental conditions applied.

Table 2 also presents data relating to the processing time of the CIL. The average processing time for the CIL was consistently around 0.15ms across all experiments conducted.

Table 2. Key experimental statistics

	Exp. 1 – Autopilot Trial 1	**Exp. 2 – Roll Trial 7**	**Exp. 3 – Pitch Trial 2**	**Exp. 4 – Yaw Trial 5**
Length Of Trial (h:m:s.ms)	0:05:03.219	0:00:35.265	0:03:29.422	0:04:04.953
Total Number Of Data Packets	13871	1445	9201	10639
Average UDP Rate (packets/second)	45.55604296	40.94710295	43.93043497	43.40424311
Processing Time				
Average CIL Processing Time (ms)	0.154797195	0.131487889	0.156939463	0.156023700
Average Combined CIL And Visual Model (ms)	0.183015999	0.197231834	0.170524943	0.201542368
Roll				
Average Error Between Simulation Environment and CIL (°)	0.000000092	0.000001801	0.000001286	0.000001145
Pitch				
Average Error Between Simulation Environment and CIL (°)	0.001392787	0.017359439	0.085097361	0.049037612
Yaw				
Average Error Between Simulation Environment and CIL (°)	0.000002413	0.000002335	0.000002171	0.000002085

The average time between data arriving at CIL and arriving at the visual model ranges from 0.17ms to 0.20ms. With this in mind, examination of Table 1 reveals that the times recorded in all three system modules are identical. Hence, total processing time is clearly less than 1ms. As time was measured to millisecond accuracy calculations should then show that this transition takes no time to occur. An average processing time of 0.15ms can be explained by referring to the second record in Table 1. When examining the highlighted results it can be seen that on this occasion the CIL actually took 16ms to process the data received from the simulation software and send the updated joint angles to the visual model. Analysis of the complete results from the various trials showed that this exception occurs periodically in the CIL. Hence, when averaged over the total number of packets processed the quoted 0.15ms processing time is found. A periodic processing time of 16ms in the CIL is obviously longer than necessary, when it can clearly operate in less than 1ms. This exception could be due to the program periodically flushing data buffers or it could even be due to a periodic demand on the computer from other processes that it is executing. Additional tests are currently being conducted to determine the source of this timing discretion. However, it is very promising to see that the CIL can introduce a range of extra features to the UMS but still perform its primary task in less than a millisecond.

The remainder of Table 2 refers to the communication of the roll, pitch and yaw variables. The error between the values commanded at the simulation software and the values received at subsequent stages in the system is extremely small. For example, during Experiment 2 the average error in the roll variable between the simulation software and the visual model was less than 0.000002 of a degree. Looking across the various experiments listed it can be seen that these results are consistent for the autopilot, pitch and yaw trials. The average error in the yaw variable is also of a similar magnitude. However, it is clear that the average error in the pitch variable is significantly larger. While the error is still less than 0.1°, there is no reason why this variable should behave any differently to the roll and yaw. Following the experiments conducted the CIL program was thoroughly inspected and the cause of this problem was identified and will be promptly addressed. Hence, when rectified the CIL will enable the UMS to reproduce the motion of a simulated vehicle to within a fraction of a degree in all variables concerned.

CONCLUSIONS AND FUTURE WORK

In this paper we have presented a novel training simulator that employs a robotic arm to overcome the limitations of traditional motion simulator designs. A key component in the flexible and modular design of this simulator is the Control Interface Layer. This simulation-control layer enables a realistic simulation environment to be interfaced with the novel simulator platform or a virtual model. Unlike existing designs, this allows the motion simulator to accept input from not only the user, but the environment with which they are interacting also, facilitating realistic high-fidelity motion enabling the user to be fully immersed in the simulation. The presented Control Interface Layer enables the simulator to follow the motion of a simulated vehicle with fine precision while introducing minimal latency within the control loop.

The capabilities of the control layer described in this paper may be extended so that it can be interfaced with a range of different vehicle simulations. In the future we plan to conduct extensive testing and validation of the system with the simulator hardware. The development of a realistic helicopter cabin or pod to be attached to the simulator is an ongoing focus, as is the design of the motion cueing algorithm for the Universal Motion Simulator.

REFERENCES

Aponso, B.L., Beard, S.D. & Schroeder, J.A. (2009). The NASA Ames Vertical Motion Simulator – A Facility Engineered for Realism. *Proceedings of the Royal Aeronautical Society Spring 2009 Flight Simulation Conference.*

Beykirch, K, Nieuwenhuizen, F.M., Teufel, H.J., Nusseck, H.G., Butler, J.S. & Bulthoff, H.H. (2007). Control of a Lateral Helicopter Side-step Maneuver on an Anthropomorphic Robot. *Proceedings of the 2007 AIAA Modeling and Simulation Technologies Conference*, Vol. 2, pp. 1010-1017.

Cremer, J & Papelis, Y (1993). The Software Architecture for Scenario Control in the Iowa Driving Simulator. *Proceedings of the 4th Computer Generated Forces and Behavioral Representation Conference*.

James, M.R., & Negaard, G.R. (1981). *Structural Analysis of the LAMARS Assembly*. Aerospace Structures Information and Analysis Centre, Wright-Patterson Air Force Base, Ohio.

Kecskeméthy, A, Masic, I & Tändl, M (2008). Workspace Fitting and Control for a Serial-Robot Motion Simulator. *Proceedings of the Second European Conference on Mechanism Science*, Vol. 1, pp. 183-190.

Niccolini, M, Pollini, L, Innocenti, M, Giordano, P.R., Teufel, H.J. & Bulthoff, H.H. (2009). Towards Real-Time Aircraft Simulation with the MPI Motion Simulator. *Proceedings of the 2009 AIAA Modeling and Simulation Technologies Conference*.

Nusseck, H.G., Teufel, H.J., Nieuwenhuizen, F.M. & Bulthoff, H.H. (2008). Learning System Dynamics: Transfer of Training in a Helicopter Hover Simulator. *Proceedings of the 2008 AIAA Modeling and Simulation Technologies Conference*.

Schaetzle, S, Preusche, C & Hirzinger, G (2009). Workspace Optimization of the Robocoaster Used as a Motion Simulator. *Proceedings of the 14th IASTED International Conference on Robotics and Applications*, Vol. 1, pp. 470-477.

Stewart, D. (1965). A Platform with Six Degrees of Freedom. *The Institution of Mechanical Engineers, Proceedings 1965-66*, Vol. 180, Part 1, No. 15, pp. 371-386.

Teufel, H.J., Nusseck, H.G., Beykirch, K.A., Butler, J.S., Kergers, M & Bulthoff, H.H. (2007). MPI Motion Simulator: Development and Analysis of a Novel Motion Simulator. *Proceedings of the 2007 AIAA Modeling and Simulation Technologies Conference*, Vol. 1, pp. 335-345.

Wentink, M, Bles, W, Hosman, R & Mayrhofer, M (2005). Design & evaluation of spherical washout algorithm for Desdemona simulator. *Proceedings of the 2005 AIAA Modeling and Simulation Technologies Conference*, Vol. 2, pp. 1201-1211.

White, M.D. & Padfield, G.D. (2004). Flight Simulation in Academia: Progress With HELIFLIGHT at The University Of Liverpool. *Flight Simulation 1929-2029: A Centennial Perspective, The Royal Aeronautical Society Flight Simulation Conference 2004.*

Zheng, S, Zheng, S, He, J & Han, J (2008). An Optimized Distributed Real-Time Simulation Framework For High Fidelity Flight Simulator Research. *Proceedings of the 2009 International Conference on Information and Automation*, Vol. 1, pp. 1597-1601.

Zheng, S, Huang, Q, Jin, J & Han, J (2009). Flight Simulator Architecture Development and Implementation. *Proceedings of the 2009 International Conference on Measuring Technology and Mechatronics Automation*, Vol. 2, pp. 230-233.