Implementing Precision Tracking within a Distributed Simulation Environment: Overcoming Limitations

Mr Lee Blucher; Mr Bjorn Wharff
QinetiQ Pty Ltd
lblucher@qinetiq.com.au; bjwharff@qinetiq.com.au

Abstract. Distributed simulation provides significant advantages to the modelling and simulation (M&S) community by promoting component interoperability and reuse, whilst facilitating computational load balancing across multiple computing resources comprising the simulation infrastructure. As such, large scale and high resolution simulations become possible by compartmentalising and interconnecting smaller simulations without the need to invest substantial funds into centralised supercomputing facilities. Distributed simulation itself introduces many complexities, primarily the transfer of appropriate data of adequate resolution and frequency. Several prevalent distributed simulation protocols exist, each attempting to overcome the inherent problems associated with distributed simulation.

This paper refers to a Distributed Simulation Environment (DSE) which utilises the Distributed Interactive Simulation (DIS – IEEE 1278) protocol to interconnect individual simulation components to form high level simulations. Different components of the simulation require processing at varying levels of fidelity and as such difficulties arise with the resolution of data and methods of describing the simulation interactions. Overcoming these problems requires significant forethought and planning to achieve the fidelity requirements using a protocol which may not be best suited to the individual needs of the model.

This paper discusses the approach used to implement a precision tracking algorithm into a DSE in an attempt to overcome the inherent problems introduced by distributed simulation and the DIS protocol specifically. A series of observed shortcomings will be identified in serving the needs of the algorithm and potential solutions discussed, providing insights and reasons for adopting those solutions. It is expected that the lessons learnt in this implementation effort will benefit the M&S community actively involved in Distributed Simulation.

1. INTRODUCTION

Modelling and simulation (M&S) is increasingly being used by Defence to assist with Research and Development (R&D); it is acknowledged as a powerful and resource efficient approach for exploring new concepts, providing training aids, saving resources, and reducing risk.

One branch of M&S is Distributed Simulation, which is carried out within a Distributed Simulation Environment (DSE), and brings many advantages to the simulation community. This paper is based on a DSE that utilises the Distributed Interactive Simulation (DIS – IEEE 1278) protocol, which enables the connectivity of multiple simulation systems to create high level simulations.

1.1 Distributed Interactive Simulation

The Distributed Interactive Simulation protocol is used by the M&S community as an open standard for conducting real-time platform-level war-gaming across multiple host computers; this is termed Distributed Simulation. The power of this protocol lies in the select way in which it uses pre-defined packet formats to share information about entities and interactions in a simulation with connected participants, minimising the volume and frequency at which the data are shared. The most powerful attribute of the protocol is the concept of ‘dead reckoning’, which allows pre-defined thresholds to drive the update rates of data shared between participants.

The value of DIS is in interconnecting individual simulation systems or facilities to form scalable and high-resolution simulations, without the need to centralise computing infrastructure, which in some cases is cost inhibitive or physically impossible. Reducing the impact of geography further empowers each participant to supplement on-site with off-site capability, maximising the re-use of established technology and capability.

The benefits of distributed simulation do not always outweigh the costs that are introduced, and therefore should be assessed on a case-by-case basis; inevitably what will satisfy one situation may not satisfy another and in fact may compromise the results. The benefits of bringing models and simulations together are obvious; however the costs are not always apparent and are sometimes disguised. These costs were observed during the implementation of a precision tracking algorithm within a DSE; the outcome of which has been a series of methods and techniques employed to off-set the costs introduced by distributed simulation.

1.2 Environment

The Defence Science & Technology Organisation (DSTO) operates a simulation facility referred to as the Survivability Integration Laboratory (SIL), which is built upon a Distributed Simulation Environment...
(DSE). The DSE is host to many individual models and simulations which are interconnected using the DIS protocol, Virtual MIL-STD 1553 and hardware-in-the-loop to assist in the assessment of aircraft survivability and countermeasure effectiveness for the Australian Defence Organisation (ADO).

Specifically, the DSE provides a scalable and dynamic architecture which supports:

- Multi-fidelity modelling;
- Integration of hardware-in-the-loop, human-in-the-loop and modelling & simulation;
- Interoperability (interaction between models i.e. stimulation of sensors);
- De-centralised processing and distributed modelling; and
- Real-time processing.

Presagis’ STAGE war-game is used as the primary DIS-enabled simulation controller, providing synchronisation and environment definition (entity/platform locations and behaviours) to the other simulation participants. Platform and sensor modelling is mostly undertaken externally to the STAGE application, using DIS for interoperability. External models are preferred due to the capacity for increased fidelity, decentralised processing and load balancing across computing infrastructure.

An individual model (of a system) is seemingly able to attach to another entity within the simulation and interact with the environment using the DIS Protocol Data Units (PDUs). This interaction can be purely passive (receive only), or may be active through transmitting PDUs detailing the effects they have on the environment (e.g. changes of position, or transmission of Radio Frequency (RF) energy). Other models within the DSE interpret any potential effect on them and respond appropriately. E.g. the firing of a missile would result in a threat warning from a Missile Approach Warning Systems (MAWS) which the pilot would subsequently respond to using an appropriate countermeasure or tactic.

The DSE uses a number of Electronic Warfare Self Protection (EWSP) models that characterise sensors onboard operational aircraft. These sensor models use DIS to listen and determine whether detections and/or identifications have been made, based on various stringent criteria (closely based on the system, environment and scenario).

1.3 The Model

To supplement existing EWSP capability, design and development of a Directed Infrared Countermeasure (DIRCM) system was undertaken in accordance with DSTO requirements for the SIL.

The DIRCM system consists of an IR Jam Head and Laser. The IR Jam Head is a gimbal-mounted system that has movement in azimuth and elevation. In the real-world, the system is attached to the host platform, and interfaces with a MAWS to receive detection and declaration data for incoming threats. This information is used to instruct the IR Jam Head to acquire and defeat incoming threats, using initial azimuth and elevation information from the MAWS (and later using its own IR tracking sensor), in conjunction with the aircraft Inertial Navigation System (INS) data; this allows the DIRCM to rapidly acquire and maintain target track with changing aircraft dynamics. Figure 1 shows a schematic of the simulation interactions.

![Figure 1: Schematic of the simulation interactions](image1)

To model the movement mechanisms of the IR Jam Head accurately a high precision tracking algorithm was required. The tracking algorithm is used to move the azimuth and elevation gimbals independently, and to maintain the threat as close to the bore-sight as possible. Figure 2 shows two images; one is a schematic of an IR Jam Head, indicating the two axes of rotation, while the other provides an example of the physical appearance of a Jam Head.

![Figure 2: IR Jam Head](image2)

2. PERFORMANCE REQUIREMENTS

For the DIRCM model to operate effectively within the DSE, a number of performance requirements needed to be met during the implementation of the model. These were:

- Real-time performance due to human-in-the-loop and hardware-in-the-loop limitations;
- Calculation of positional updates at a rate greater than 300 Hz;
• Algorithm state information had to be accessible (for use in dead-reckoning calculations);
• Independent modelling of the azimuth and elevation gimbals;
• Maintain less than 0.2 degrees tracking error off bore-sight for stable (locked) incoming threat engagements.

3. PRECISION TRACKING ALGORITHM
In order to model the motion of a DIRCM System, a precision tracking algorithm was selected that was easily translated into C++. This section provides background information on the tracking algorithm and the mathematics behind the system to enable modelling within a DSE, and also describes the characteristics of the algorithm that assists in making comparisons against the performance requirements outlined in Section 2.

3.1 Background
The tracking algorithm is based on a Type II Control and Servo design (Reynolds, 2007) which models the rotation of a gimbal in terms of either the azimuth or the elevation. Two instances of the algorithm were used to model the azimuth and elevation gimbals concurrently but independently. To enable effective tracking of a moving target, the algorithm was required to include its state information (current gimbal velocity and acceleration data).

Figure 3 demonstrates that the tracking algorithm is a feedback control system that uses the output from the previous calculation, in conjunction with state information, to determine future positions. In order to drive the algorithm, it requires (at a minimum) the desired pointing angle as input to the system.

The initial velocity and position of the system may also be supplied to the algorithm as initial conditions; this capability is used to overcome limitations that are identified later in this paper.

![Figure 3: Type II tracking algorithm](image)

3.2 Algorithm Mathematics
The tracking algorithm identified in Figure 3 comprises compensation gains, a transfer function, two integrators, and a number of limiters. Since the algorithm is easily represented within a Matlab™/Simulink™ model, the initial testing of the algorithm’s conformance with the performance requirements was conducted within this environment.

Shown below is the definition of the transfer function \( G(s) \) indicated in Figure 3; its derivation is detailed by Reynolds (2007). The output of \( G(s) \) is an acceleration vector, which is limited to the system performance. The acceleration is integrated to obtain the angular rate (or velocity), which again is limited and integrated to provide the next position in the time series. Finally, the position is limited by the physical extents of the system.

\[
G(s) = \frac{(st_1 + 1)}{\frac{1}{w_0^2} \left(1 + \frac{w_0^2}{1 + w_0^2} t_2 \right)^\frac{1}{2} \left(s + \frac{1}{w_0^2 t_1}\right)}
\]

where \( t_1 = \frac{\tan(\phi_m) + \sec(\phi_m)}{w_0} \), and \( t_2 = \frac{1}{w_0^2 t_1} \)

From looking at the transfer function \( G(s) \), it is clear that there are only two configurable parameters, the phase margin \( \phi_m \), and the system bandwidth \( w_0 \) (both of which affect the stability and response of the system).

4. MODEL IMPLEMENTATION
The development of the DIRCM System within a DSE required the implementation of the tracking algorithm, such that it operated in the same manner as in the Matlab™/Simulink™ model. During development, performance limitations were encountered, and methods to overcome these limitations were addressed.

4.1 Algorithm Implementation
To enable the implementation of the algorithm in the C++ language, the transfer function \( G(s) \) required translation into its state-space representation (shown in the equation below). To achieve this, \( G(s) \) had to be solved initially by selecting values for both the phase margin \( \phi_m \) and the system bandwidth \( w_0 \) and then translated to the state-space representation using the applicable Matlab™ command.

\[
\frac{dx}{dt} = Ax + Bu
\]
\[y = Cx + Du\]

Where \( A, B, C \) & \( D \) in the equations are solved using Matlab™.

The integrator blocks were also converted to their respective state-space representations, and incorporated into the DSE model.

4.2 Primary Performance Considerations
Given the DSE in which the system is to function, there were various primary performance considerations that were addressed during the design and development process; specifically, these included the effect that dead-reckoning of simulated entities had on the gimbals of the DIRCM system, and problems associated with beam representation for receiving entities (sensors which would be affected by DIRCM emissions).
4.2.1 The Effects of Entity Dead-Reckoning on Azimuth and Elevation Angles

To ensure that models within a simulation use identical data, each model maintains a dead-reckoned representation of the entities of interest. As the dead-reckoned parameters exceed the actual (truth) data by pre-defined thresholds, an Entity State (ES) PDU is sent from the source entity, and each model maintaining track of the entity updates its dead-reckoned parameters. As a result, when the ES PDUs are processed, the dead-reckoned state data of the entity are instantaneously updated to the latest truth data, resulting in the entity potentially having a new position and orientation.

Given that the tracking algorithms are calculating threat azimuth and elevation angles in the host-platform’s body coordinate system, the receipt of an ES PDU for either the host platform or the target entity may result in the target being in a changed apparent position relative to the host platform.

Figure 4 highlights this problem in a simplified two-dimensional case by displaying the elevation angle (a) of a beam pointing at a target before an ES PDU is received for the host platform, and after the ES PDU is processed. The only parameter that has been updated in the ES PDU is the host platform’s pitch angle. It shows that the beam’s elevation angle has remained constant (in the host’s body coordinate system), and the target is now no longer within the beam; the discrepancy is exaggerated in Figure 4 for demonstration purposes.

This situation is only introduced due to the distributed nature of the environment using dead-reckoning, but is a situation that must be overcome to achieve the system’s performance requirements.

4.2.2 Beam Representation

The aim of the DIRCM system is to direct energy at a threat, in an attempt to affect its navigation system, and ultimately avoid an intercept. Due to the narrow beam divergence of the directed energy being emitted, and the nature of the environment, difficulties arise in the simulated entities maintaining an accurate representation of the beam.

Each entity in the environment that may be affected by emissions must maintain a dead-reckoned representation of each beam to determine whether the beam is incident on its sensor. Given the narrow size of the beam, any small angular changes in the beam’s actual position (as compared to the dead-reckoned beam representation) could result in a simulated entity falling outside that beam.

In addition to the issues presented above, the DIS protocol does not contain a PDU that facilitates the dead-reckoning of directed emissions.

4.3 Analysis and Overcoming Limitations

The primary performance considerations identified in the previous sub-section had to be addressed and overcome in order for the tracking algorithm to be used in the DIRCM system. Each of the considerations identified are addressed in the analysis, which includes the methodology and design decisions used to overcome the limitations.

4.3.1 The Effects of Entity Dead-Reckoning on Azimuth and Elevation Angles

The necessary process of dead-reckoning entities within a DSE results in the performance issues identified in section 4.2.1. To overcome these issues, a number of techniques were devised to minimise the inherent errors.

Prior to processing a host-platform or target entity ES PDU, the difference between the actual line-of-sight (LOS) angles (in azimuth and elevation) and the current beam pointing angles are saved in memory. After processing the ES PDU, the new LOS angles are calculated, and the beam pointing angles are offset by the saved values. Applying this position-offset technique ensured that any major positional or orientation changes as a result of ES PDU updates were nullified.

A recorded simulation of a host platform targeting a threat was run twice; Figure 5 highlights the errors that were introduced when ES PDUs were received over an engagement period; of particular note are the spikes that signify a processed ES PDU, and the inherent errors that were introduced.

Figure 6 highlights the reduction in error when applying the position-offset technique. Comparing the errors with the required performance threshold lines (of ±0.2
degrees), it is clear that while the errors were reduced, the performance requirements are not satisfied.

**Figure 6:** Azimuth and elevation difference with position-offset technique applied

The flexibility of the tracking algorithm allows the velocity and position of the tracking algorithm to be initialised, which was exploited to set new angular velocities for the gimbals.

The technique used to calculate the new velocities for both the azimuth and elevation beam pointing angles is similar to the position-offset technique. The velocities of both the azimuth and elevation gimbals are saved prior to processing the ES PDU, in addition to measuring the angular velocity of the threat relative to the host-platform in both azimuth and elevation. The percentage differences in the angular velocities for both axes are calculated and used once the ES PDU has been processed.

After processing the ES PDU, the data for both the host-platform and the threat are immediately dead-reckoned to project the threat and host-platform positions for the next iteration. These data are used to calculate the threat velocities (with respect to the host platform) in azimuth and elevation, and the percentage differences measured prior to ES PDU processing are applied. The calculated position and velocity are then set to the respective tracking algorithms responsible for modelling the azimuth and elevation gimbals.

The recorded simulation was run a third time, and Figure 7 indicates that the position and velocity techniques resulted in the errors being significantly reduced. It also highlights that once the threat has been engaged (and is being tracked, as indicated by the vertical lines in Figure 7), the azimuth and elevation angle errors lie within the performance thresholds. It is only at the very end of the engagement that the azimuth angle falls outside the threshold. This is acceptable given that at this point the threat is no longer flying directly toward the platform.

**Figure 7:** Azimuth and elevation difference with position and velocity techniques applied

### 4.3.2 Beam Representation

A performance consideration identified in Section 4.2.2 related to difficulties that arise with receiving entities maintaining an accurate representation of the beam. Similarly to the way in which each model maintains a dead-reckoned state of entities of interest, each model must also maintain a dead-reckoned state for emissions that may affect it.

#### 4.3.2.1 Directed EO Emission PDU

The current DIS protocol does not consist of a PDU that can be used to describe DIRCM emissions, and as such, a Directed Electro-Optic (EO) Emission PDU was defined that builds upon the existing structure of the Electromagnetic Emission PDU. Additional fields were included to the PDU for directed beam characterisation: beam angular position, velocity and acceleration, which are used to dead-reckon the beam (by receiving entities).

#### 4.3.2.2 Dead-Reckoning Parameters

The threshold chosen for dead-reckoned updates of Directed EO Emission PDUs was an important factor for two reasons: (1) Due to the narrow divergence of the beam, if the dead-reckoned angles are only slightly different to the real angles the beam may not be calculated to be incident on the target; and (2) The rate at which updates are sent may congest the simulation network.

To counter the first point, the host model calculates the beam pointing directions at a rate greater than 300 Hz, and retransmits a Directed EO Emission PDU when the beam’s azimuth and elevation angles differ from the dead-reckoned angles by greater than half the beam width. This ensures that a system dead-reckoning the beam is within half a beam width from the actual beam position. To highlight the rate at which beam pointing angles are calculated, Figure 8 shows a graph of frequency versus time for a typical engagement (the black line is superimposed for clarity; it is a moving average trend-line, with a period of 20 samples).
**Figure 8:** Frequency of beam pointing calculations

Due to the narrow beamwidths, it becomes increasingly important to maintain accuracy within the beam pointing function. Figure 9 shows a 3-dimensional graphic (produced using SIMDIS™) of the system directing a beam at an oncoming threat. At a range of 1.5 km the diameter of the beam is marginally larger than the target. This demonstrates that minor beam pointing errors can affect the on target time of the DIRCM, and consequently the effectiveness of the countermeasure.

**Figure 9:** A threat depicted inside a directed beam at 1.5km

5. **CONCLUSIONS**

In this paper we have discussed possible approaches to be used when implementing a precision tracking algorithm to overcome the inherent problems associated with high resolution modelling within a DSE.

The primary issues that were addressed were: (1) The effect that dead-reckoning of simulation entities have on the performance of a precision tracking algorithm; and (2) The problems associated with simulation entities representing the beam emitted by the model.

Through the use of state information available from the precision tracking algorithm (position, velocity and acceleration data), solutions to the identified primary issues were developed. Analysis of the methodologies used indicated that the final performance of the precision tracking algorithm satisfied the model’s requirements.

Some inherent errors still exist in the system. These are unavoidable due to the distributed nature of the environment, and the characteristics of the tracking algorithm itself. Over time, methods to reduce these errors further will be investigated, but it is envisaged that the lessons learned and approaches discussed in this paper can benefit the M&S community concerned with implementing precision tracking algorithms or other high-fidelity motion models in a DSE.

**REFERENCE**