A NETWORK APPLICATION INTERFACE FOR RCS CALCULATIONS

Philip E. R. Galloway

Roke Manor Research Ltd,
Old Salisbury Lane, Romsey, Hampshire United Kingdom.
`philip.galloway@roke.co.uk`

***Abstract*— This paper describes a Network Application Interface, Network (API) that has been added to the Epsilon^{TM} Radar Cross Section (RCS) Prediction Code. Epsilon^{TM} is a well established Physical Optics (PO) based prediction code with Shooting Bouncing Ray (SBR) multiple scattering and Physcial Theory of Diffraction (PTD) edge scattering facilities. The Network API is a recent addition to the product that allows users to utilise the RCS prediction engine into their own applications. The main use of this facility is anticipated to be for the incorporation of accurate RCS data into simulation frameworks. We describe the Network API, and provide an example of how the API has been used within a simulation of a Multi-Static Passive Radar (MSPR) system.**

I. INTRODUCTION

The Epsilon™ RCS prediction code was initially developed by Roke Manor Research in the mid 1980s to support a requirement for platform signature control within the UK Defence market. It is an export controlled application, and is used by organisations in several NATO countries in addition to the UK. Previous conference papers [1]-[4] chart the history of developments in the code. This paper will concentrate on the Network API that has been built to allow the incorporation of direct RCS predictions using Epsilon™ into third party simulation applications.

The first section details the motivation for the development of the facility and discusses the benefits foreseen over other standard approaches to this problem domain; we then provide a detailed description of the API. We follow this with details of how the API has been incorporated into a simulation of a Multi-Static Passive Radar system and we present some comparative results for simulations undertaken with and without directly predicted RCS data. Finally we summarise the capability, its limitations and potential benefits to our user base.

II. RATIONAL

The design of radar systems of all types depends on the RCS of the objects that are intended to be detected and also on the RCS of objects that also appear in the radar field of view and are perhaps undesirable sources of reflection. One of the specifications of radar performance often stipulated is the probability of detection for a fixed value RCS target at a particular range. This metric allows the designer to determine the required power on target, dwell time etc. in order to ensure a sufficient signal to noise ratio at the receiver to meet the detection requirements[1]. The behaviour of targets in the real world is that there is practically never a constant RCS present, as this figure is dependent on many factors and typically varies very rapidly with time when relative motion of the radar and target is involved. Radar design processes that adopt this simple approach have historically been quite successful and there probably has been less attention paid to the complexity of target scattering properties than is perhaps deserved.

It should be noted that the problem with target scattering variation has historically been addressed by using statistically derived Swerling models [5], however for the purposes of this discussion the aim is to use causal data within the simulation.

Recently there have been a number of developments that have required some improvements to the traditional Radar design process; the most influential of these is perhaps the development of Stealth platforms, however there is also increasing pressure on the allocation of the Radio Spectrum where it is desirable that less space is used for radar systems.

---

[1] This is a very simplified view of Radar System Design and is used in this context in order to highlight the possible benefits of considering RCS in more detail during the Radar design cycle. The interested reader should consult Skolnick's book [5] for an excellent introduction to the subject.

There is also a significant change in the technology base for signal processing and network infrastructure that opens up many possibilities for novel architectures for Radar systems. In this context, the simulation of Radar system performance using synthetic environments is becoming increasingly common and growing in sophistication. There is an emerging demand to be able to feed these simulations with high quality RCS data and it is for this reason that we have added a network Application Interface (API) into the Epsilon™ product.

A. *Alternative Approaches*

Almost all synthetic environments dealing with the interaction between radar systems and targets will have target RCS models that are of the "Single Point Target constant RCS" type. The qualities of this type of approach are summarised in Figure 1.

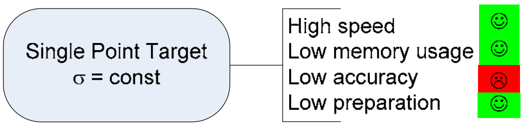

Figure 1 Single point constant RCS model qualities.

The big problem with this approach is the Low accuracy which has been addressed with other approaches. One step up in sophistication from this would be RCS models based on lookup tables that provide RCS as a function of incident angle. Lookup table approaches can source their data from measurements or predicted data, but always have a limitation on angle resolution, which for electrically large targets means that the scintillation rate of the target is not captured accurately. If wide bandwidth data is needed then the size of the lookup table may become prohibitive. The qualities of this type of approach are summarised in Figure 2.

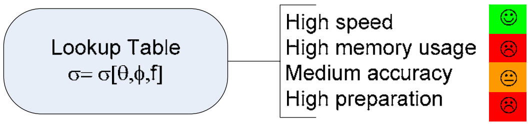

Figure 2 Lookup Table RCS against Aspect and Frequency qualities.

One approach to mitigate this problem is to use a multi-point scattering model approach, where the RCS characteristics can be emulated by a small set of discrete scatting centres. Exactly how the positions of the scattering points and amplitudes is derived is often unclear, however one approach has been reported on in a paper by Simpson et al [6] that employs a super-resolution technique to derive an equivalent multi-point scattering model directly from predicted data. The qualities of this type of approach are summarised in Figure 3.

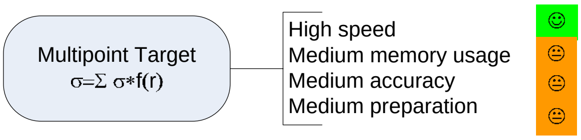

Figure 3 Multi-point Scattering model qualities.

These alternative approaches suffer from simplification and approximation and may cast doubt on the findings of any simulation based investigation.

B. *Benefits of the Network API*

A network API provides RCS signature data as a function of aspect to the resolution of the numerical representation of the aspect description in the request message. This is significantly superior to a lookup table approach in that to provide a similar resolution would need both a vast memory to store the lookup table in the simulation and a significant prediction resource to create the data for the lookup table.

A network API provides RCS prediction data at exactly the aspect required, this is superior to the equivalent multi-point model approach that relies on the equivalent multi-point model being valid over a range of aspects. The Network API approach also offers the capability to predict wide bandwidth signatures, allowing for example the accurate simulation of the video signal within the radar model. The prospect of attempting this functionality from a lookup table approach is daunting and often hundreds of data points are then needed for every aspect that is captured in the table. A multi-point scattering model approach can accommodate this level of functionality, however the approximations are stretched further and accuracy will suffer. The qualities of this type of approach are summarised in Figure. 4.

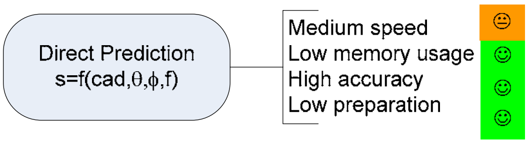

Figure 4 Direct Prediction using Network API qualities.

C. Limitations of the Network API

Currently the processor loading to make a single point prediction is significantly longer than that for a lookup table or a multi-point model approach. This means that for simulations where a large number of predictions are required, runtimes will increase. As the API runs over a network link this extra compute load can be undertaken in parallel by a dedicated prediction computer, however it is anticipated that this will remain an issue for some time to come. If a simulation requires many different target types to be modelled simultaneously then multiple Epsilon™ licenses would be required, making an approach of this kind a significant decision in terms of cost.

III. NETWORK API DESCRIPTION

The Network API is composed of a C/C++ dynamic link library, header files and a full copy of the Epsilon™ software. The software distribution for Epsilon™ Release 16.0 comes with a worked example client that can be built using the Microsoft Visual Studio Integrated Development Environment (IDE). The API is reasonably simple and follows a familiar C++ based “Construction”, “Activity”, “Destruction” form. The calls to the API are listed in Table I in the order that they would typically appear within a client.

TABLE I
FUNCTIONAL INTERFACE OF NETWORK API

Function	**Description**
ecl::Create()	This function instantiates the interface and returns a handle to the created client. Multiple interfaces are supported, each with a unique handle. All other calls to the interface expect a reference to this handle.
ecl::Init()	This function initialises the client. It must be called once after the call to the Create() function has returned a handle to the application.
ecl::Connect()	This function attempts the connection to the Epsilon™ prediction engine.
Ecl::SetScript()	This function specifies the prediction script that will be used by the Epsilon™ prediction engine. This is part of the initialisation phase and must be called before any call to Process()
ecl::Process()	This function accepts an RCS datum request from the client, runs the prediction and returns a result data structure.
ecl::Disconnect()	This function disconnects the client from the Epsilon™ prediction engine.
ecl::Uninit()	This function is called as part of the client de-allocation sequence. It should be called after the Disconnect() call and before the Destroy() call.
ecl::Destroy()	This function effectively de-allocates the client.

The Create(), Init(), Connect() calls set-up the network link to the Epsilon™ solver running as a separate process on either the local machine or a remote machine. The start-up of the Epsilon™ solver instance is not automatic and can either be managed by the client using system calls, or manually started up. The Epsilon™ solver instance must have direct access to the script file specified in the SetScript() call, typically this will reside in the local directory where the solver was started. The main interface between the client simulation and Epsilon™ is via the Process() function call. This accepts an RCS request data structure that specifies the geometrical orientation of the model-relative to the radar and returns a result data structure with the RCS in the form of a scattering matrix.

A. RCS Request data content

The request data is held in an ecl::ecl_aspect structure which contains the description of the relative positions and orientations of the transmitter, receiver and target. In addition the frame of reference for the result data is defined by specifying the transmitted electric field vector direction. The API assumes a linear polarisation is transmitted, but returns a full scattering matrix so that the client can, if needed, evaluate the scattering from any arbitrary elliptical polarised illumination wave.

B. RCS Result data content

The result data is held in three orthogonal components within the target's illumination frame of reference. The scattering results are aligned with the illuminating E-field direction, orthogonal to the E-field direction and lastly in the wave vector direction. This third component will be identically zero for backscatter situations, but for bistatic angles will be non-zero.

IV. AN EXAMPLE MSPR SIMULATION

A Multi-Static Primary radar system consists of a set of geometrically separated transmitters and receivers that co-operate together to act as a radar. The arrangement of the transmitters and receivers is an important factor for the modelling of the system and in our example we have chosen a hexagonal cell arrangement for simplicity. This cell arrangement is shown in Figure 5. The provision of RCS signatures for such a simulation is quite demanding because both transmitter position relative to the target, and receiver position relative to the target can vary independently and a pre-computed lookup table of any detail would be impractical because of the very large number of data items to be stored. A multi-point scattering model would also run into problems if computed for backscatter and used for bistatic RCS, so a direct prediction approach is a very attractive solution.

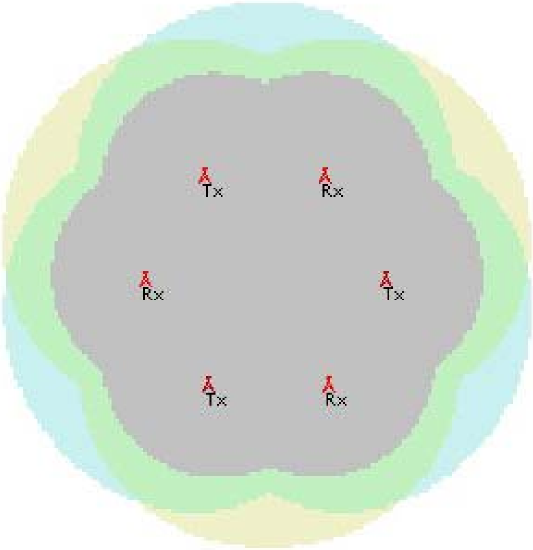

Figure 5 Typical; MSPR Deployment with coverage visualisation.

In our example simulation we select a single Transmitter Receiver pair and calculate the received signal power at the receiver generated by an example target as it moves within the coverage region. The track selected is at a constant altitude of 500m above a flat ground plane. The duration of the track is 40s and the speed of the object is approximately 275 Knots. The simulator used for this example is an in-house general purpose Multi-Static Radar Modelling Tool (MSRMT).

The MSRMT was created to support development work for multi-static radar systems. It is a windows application written in Visual C++ using the MFC libraries. The core capability is the synthesis of received video data in a synthetic receiver within a synthetic environment consisting of transmitters and reflection sources. This takes the form of an event driven simulation and incorporates:

- Free space Link budget.

- Doppler shift.
- Realistic Bistatic RCS capabilities by linking to Epsilon^{TM}.

The MSRMT also has a number of ancillary capabilities for supporting analysis and trials activities, including:

- Scenario planning.
- Synthetic track generation.
- Volkslogger (GPS position logging unit) data decoding. (http://volkslogger.de)
- GSM/GPS survey data decoding and display.

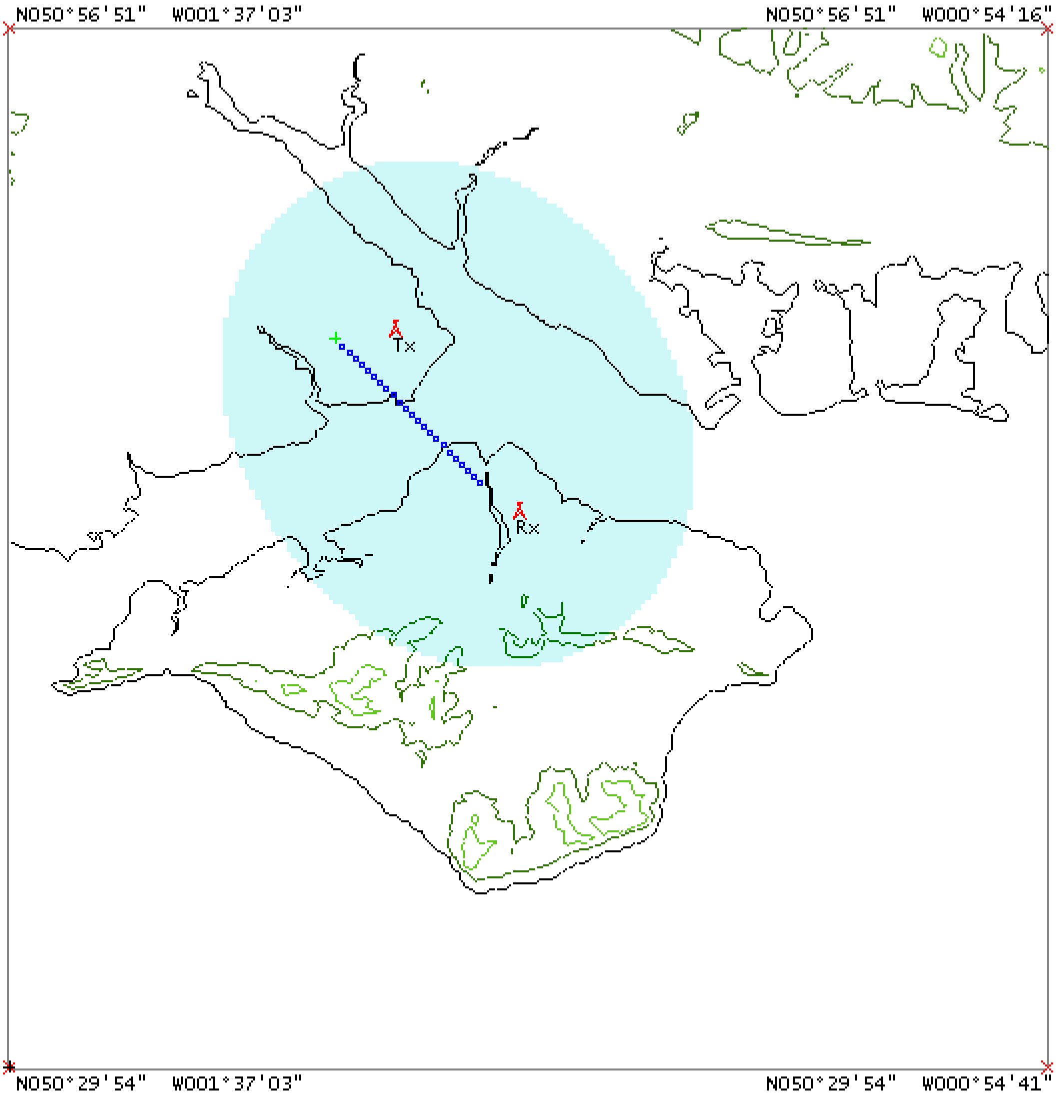

Figure 6 Simulation Deployment (MSRMT scenario planning screenshot)

Three target types have been selected for this example to illustrate the consequences of the simplifications being made with the choice of RCS model. The first model is a 0 dBsm omni scattering target, this provides a baseline for comparing the simulated data from the other target types. The second target is a 1m radius sphere (π m^2 backscatter) RCS, which illustrates the complexity involved in modelling a Bistatic scattering scenario. The third target is a Piper Warrior light aircraft model, a fairly small aircraft target shown in Figure 7.

Figure 7 Piper Warrior CAD model

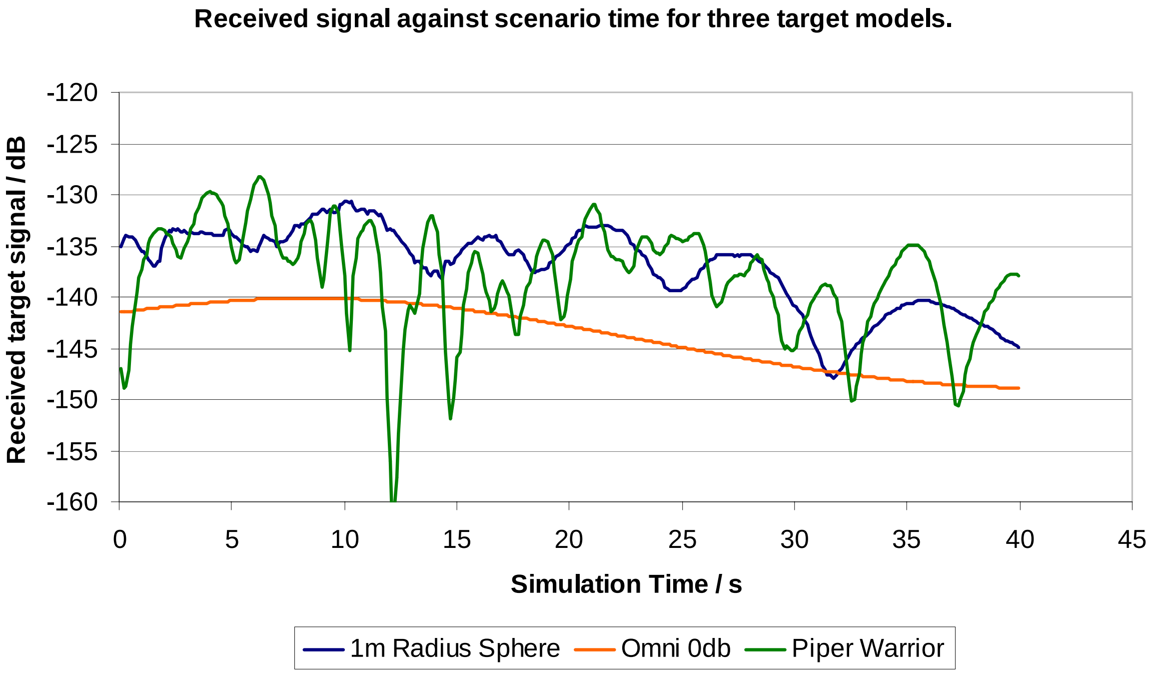

Figure 8 Received signal power against scenario time for three model types.

The Omni 0dB line in Figure 8 shows the characteristics of the received power due to the attributes of the radar geometry and antenna beam patterns. The dark blue line is a 1m radius sphere (an almost omni scattering target) but as the bistatic angle is more than ninety degrees the signature variation adds a slow ripple of about 5dB. The Piper Warrior model shows a higher rate of signature variation during the track and it should be highlighted that this is in the absence of modelling any target roll pitch or yaw orientation changes.

The example simulations were performed using a Dual Xeon 2.2 GHz Windows 2000 SR4 platform with 1GBytes of memory. The performance metric is simply elapsed time to perform the 40s simulation. The results of the performance are shown in TABLE II. As the Epsilon™ prediction runs in a separate process the simulations that use Epsilon™ are dominated by the Epsilon™ prediction times, however the use of Epsilon™ is not prohibitive with the simulation times being extended by about a factor of two in the case of the Piper Warrior model.

TABLE II
PERFORMANCE FOR THREE MODEL TYPES

Simulation	**Simulation Duration / s**
Omni 0db Target (built in scattering model within MSRMT).	765
1m Radius Sphere	875
Piper Warrior	1455

As the Epsilon™ resource can utilise multiple PCs to spread the work load then in principle the simulations could be run with minimal impact on the underlying simulation program.

V. CONCLUSIONS

This paper has presented the Epsilon™ network API as a possible solution of adding high accuracy RCS data into the simulation of modern radar systems. A specific example of a Passive MSPR model has been described and some simulated signatures have been presented.

We have demonstrated that implementing simplified target scattering models within radar simulation models will impact negatively on the validity of the simulation and have provided an illustration of some of the detailed effects seen when the complexity of the modelling is improved by incorporating an RCS prediction tool into an example simulation.

We have also provided some performance data that shows that the use of directly predicted RCS data need not have a significant performance impact on a modelling activity.

ACKNOWLEDGMENT

The author would like to thank Roke Manor Research Limited for the use of their facilities in the preparation of this paper.

REFERENCES

[1] M.L Harman. and S.H.W Simpson, *EPSILON - A Radar Cross Section Modelling Package*, Proc. UKSCS 1990.

[2] P.E.R Galloway and S.H.W Simpson *Parallelisation of Radar Cross Section (RCS) Prediction for Electrically Large Targets*, Proc. European Simulation Multiconference 1991

[3] S. H. W. Simpson, P. E. R. Galloway, M. Harman ,*Applications of Epsilon™ A Radar Signature Prediction and Analysis Tool*, International Radar Symposium IRS'98, Munich ,Germany, September 1998.

[4] P. E. R. Galloway., T. D. Welsh., *Extensions to the Shooting Bouncing Ray algorithm for scattering from diffusive and grating structure*s, Proceedings of International Radar Symposium IRS2005, Berlin, Germany, September 2005, German Institute of Navigation – Deutsche Gesellschaft Fur Ortung und Navigation e.V. (DGON)- Kolnstr. 70 D-53111 Bonn.

[5] M. L Skolnik., *Introduction to Radar Systems*, Second Edition, McGraw-Hill International Book Company. 1980, ISBN 0-07-057909-1

[6] S. H. W. Simpson, P. E. R. Galloway, *Extended Target Simulation at a Physical Level,* Proc. AGARD SPP Symposium on Radar Signature analysis and Imaging of Military targets, Ankara, Turkey, October 1996, CP-583