LIGHT CURVES FOR GEO OBJECT CHARACTERISATION

Emma Kerr[1], Gabriele Falco[1], Nina Maric[1], David Petit[1], Patrick Talon[1], Elisabeth Geistere Petersen[1], Chris Dorn[2], Stuart Eves[3], Noelia Sánchez-Ortiz[4], Raul Dominguez Gonzalez[4], Jaime Nomen-Torres[5]

(1) DEIMOS Space UK Ltd, Airspeed 1, 151 Eighth Street, Harwell Campus, OX11 0RL, UK
emma.kerr@deimos-space.com
(2) Inverse Quanta Ltd, Brooklands, Pankridge Street, Crondall, Hampshire, GU10 5QU, UK
(3) SJE Space Ltd, 29 Clayhill Road, Burghfield Common, Reading, RG7 3HB, UK
(4) DEIMOS Space S.L.U., Ronda de Poniente 19, Tres Cantos, Madrid, 28760 Spain
(5) DEIMOS Engineering and Systems, Calle Francia 9, 13500 Puertollano, Spain

ABSTRACT

The classification and characterisation of objects in geosynchronous orbit (GEO) is an important goal in Space Situational Awareness (SSA). Temporal variations of apparent magnitude, called a light curve, are captured by optical telescopes. Light curves contain information on features such as attitude, size, shape and materials useful for the characterisation of the object. Analysing light curves can also allow the determination of normal patterns of life, and hence the detection of occasional behaviour during the orbit, such as manoeuvres. The objective of this study is to develop a method for space object characterisation using light curves. To achieve this a high-fidelity simulator for generating photometric data and light curves is produced, and used to train the machine learning algorithm for extrapolating common features of GEO objects. This paper details the development of the simulator, and the development and initial results of the machine learning algorithm.

1 INTRODUCTION

There are many important interests in understanding and characterising space objects:

- Treaty Compliance – Clear and unambiguous information provides a clear basis for foreign policy engagement.
- Military threat assessment
- Strategic intelligence and adversary capability and intent assessment
- Debris characterisation

Light curve derived information cannot answer all of these questions, but it can provide some answers and, in many cases, highlight changes that need further investigation. Additionally, since light curve data is relatively easy and low-cost to obtain with low latency, using it to monitor object behaviour is a viable strategy. Objects that warrant further investigation can be targeted by different observational capabilities.

Apparent magnitude is a key measurement of the information that a telescope can gather about an observed space object. Analysis of the temporal variations in the apparent magnitude, (brightness), permits an analysis of the characteristics of the object.

The technique of light curve analysis is well established; an example can be seen in Figure 1, showing a plot of one hour (19:35 to 20:35 UT, 04/04/2018) of observations with the 'Tracker 2' telescope from Deimos Sky Survey (DeSS) of the GSAT satellite (18027A). More than 2600 measurements were produced during this hour, and these have been plotted to show a distinctive photometric light curve.

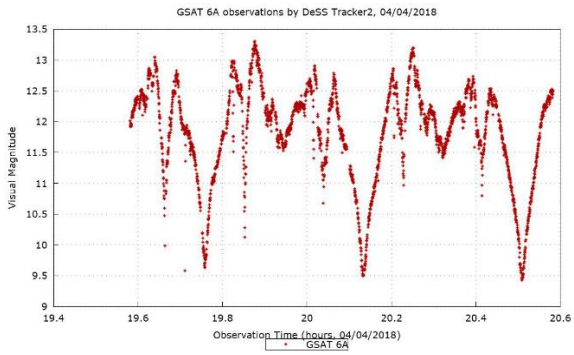


Figure 1. Light curve obtained from Tracker2 observations of GSAT satellite (18027A)

The light curve, in Figure 1, has a very well-defined main rotation period. The phase angle was quite constant, since the object was moving at quite similar apparent speed to the Sun, maintaining a similar angle with it for all the observations. The phase angle variation was calculated along the observing period at about 1°, not impacting on the observed visual magnitude. There is a clear 24-minute repetition of the pattern, so assuming the object has no clear symmetry then this gives its rotation period. If the object is quasi-symmetric then this period is doubled (48 minutes). Further observations would help to distinguish between these cases, and to establish if this is long term behaviour.

By having observations over longer periods, other angles of the satellite would be seen, building up a better model of the satellite during a particular pass. Over a period of time, this can yield a pattern of life, and a way to examine

the satellite's behaviour if this changes. Employing some simplifying assumptions and/or prior knowledge of other properties, (for example the optical properties of the object or the moment of inertia of the satellite), its operational status and changes to it can then be estimated. This sort of information has obvious tactical and strategic importance and can also be crucial for damage assessment.

Using an extensive literature review it was determined the most promising technique for analysing light curve data was through the use of machine learning techniques. Machine Learning (ML) techniques are considered to be an interesting technology to apply to the task of characterising objects through light curve information, due to their ability to identify patterns among input data.

2 STUDY LOGIC

This paper details the initial results of an ongoing three-phase study, which consists of:

- Phase 1: A comprehensive review of the use of light curves for characterisation and any issues this type of analysis may face and some possible alternative techniques.
- Phase 2: Development of a light curve simulator, development and initial testing of chosen machine learning techniques for object characterisation using light curves and a short campaign of observations using Tracker 2 at DeSS.
- Phase 3: Extensive campaign of observations using DeSS and further development and testing of the machine learning techniques.

The project is currently at the beginning of Phase 3. Hence, this paper contains the initial results gained in Phase 1 and 2 only.

3 METHODOLOGY

The approach to the characterisation of objects in space through optical observations (using information from light curves) is based on three main aspects:

- Optimisation of the light curves obtained from sensors, which shall provide the largest sampling-frequency possible, well-calibrated brightness information and high radiometric sensitivity
- Long observation arcs and repeated arcs over a number of nights throughout the year, looking for patterns in the long-term brightness data (building on the short-term approach normally used for analysing single passes)
- Machine learning techniques applied to the identification of patterns in the light curves for the classification of object geometry, attitude, materials, etc.

This project focuses on objects in the geosynchronous Earth orbit (GEO) regime, where long observations can be performed at the same elevation angle (limiting the variations in the observation conditions to known factors, such as the illumination angle of the Sun which across the year makes an apparent +/- 23.5° motion in the sky relative to the equator). However, the methods developed in this project could also be applied to characterisation of debris, assuming appropriate training data was available to the machine learning algorithms.

To increase the accuracy and the sampling frequency, and to avoid conjunctions of the targets with surrounding stars, a fast, unfiltered camera collects a series of very short images of less than 1 second exposure time. These cameras are very sensitive and do not require mechanical shutters. Other systems based on video cameras can obtain a very large sampling rate but they are not generally sensitive enough, and their data stream, if converted to single frames, would suffer a loss of resolution and linearity. The very short exposure times also have the benefit of maintaining rounded stars and not generating long trails in the image. This is essential, considering that they are needed for differential photometry once the images are compared with a high accuracy photometric and astrometric catalogue. Moreover, generating photometric measurements every second would allow the system to track several GEO targets almost simultaneously, (for example, in the case of several GEO objects operating in a cluster at a shared longitudinal slot). This particular case can present some difficulties as the observed measurements of the objects may cross and produce incorrect visual magnitude evaluations. However, appropriate filtering techniques can be applied to separate light curves from clusters.

For analysing long arcs, the objective is to track full-night light curves of geosynchronous objects, trying to analyse their behaviour not only throughout the night, but also night after night. By studying their light curves with respect to the Sun angle, it is then possible to look for variations in attitude and position over time. The Sun would slowly provide illumination angle changes, allowing further details of the satellite geometry to be determined. Moreover, the small diurnal north-south variation of the sun angle, and hence the illumination, should provide additional information. By simultaneously observing several GEO targets over a full night, it is possible to model typical geometries, and introduce these light curves into a machine-learning system, to create a model that can characterise various geometries and light curves.

Deimos owns and operates its own optical SST system located at high altitude in the Castilla La Mancha region south of Madrid. The DeSS (Deimos Sky Survey) system currently consists of 4 optical telescopes which are capable of observing objects in LEO, MEO, HEO and GEO. Tracker 2 at DeSS has been used to collect the light curves to be used in this project.

The optical tube of Tracker2 sensor is a 40cm aperture at f/8 (3251mm focal length) with focal reducer and coma corrector lenses, with a resulting focal relation of F5.5. For this setup, the final resolution is 1.20 arcsec/pixel combined with the attached CCD camera. This resolution allows astrometric measurements under arcsecond accuracy, when the signal of the satellites is spread over several adjacent pixels and the centroid is determined by subpixel value.

By comparing all illumination phases, from West to East through the night and also analysing the night-to-night light curves, it should be possible to derive much more information from the light curve. This approach allows the determination of normal patterns (which should be maintained from one pass to another) and hence the detection of anomalous behaviour.

Once the light curves are available, a machine learning technique applied to those curves allows the identification of features of the observed objects. The success of the ML is determined by the architecture of the approach and the dataset used for training the ML network. For the architecture, it is considered that a neural network approach is a good technique for the purpose of the project, but other approaches may also be investigated. The data collected from DeSS is supplemented with simulated light curves. It would be impossible to gather real data on a wide enough range of objects to the degree of accuracy required to train the machine learning algorithms adequately, thus a light curve simulator is also built within the project to produce realistic light curve data.

4 SHORT OBSERVATION CAMPAIGN

In order to properly validate the light curve simulator, it was necessary to gather a small amount of data for a real object, to test the simulator against. When the simulated light curve matches the real one, the simulator can be considered to be working properly.

For this purpose, it was necessary to choose an object with a verifiable 'ground truth', meaning the dimensions, materials etc. were well understood and could be entered into the simulator. Given that Deimos do not have assets in GEO it was necessary to pick an object operated by another entity. Inmarsat's Alphasat was selected and a data sharing agreement was put in place between Deimos UK and Inmarsat.

Alphasat is a large GEO communications satellite operated by Inmarsat and ESA. The spacecraft operates at a longitude of around 25° east with an inclination of about 0.14°. It has two deployable solar arrays spanning approximately 40m, and an approximately 12m wide antenna reflector. The orientation of the satellite solar arrays is approximately North-South. The solar panels rotate to track the Sun.

Figure 2 shows an artist impression of Alphasat in orbit, it should be noted that the solar sailing tabs (the small tabs on the solar arrays) were not included in the final design of the satellite. Also, of particular note is that the reflector is a mesh, not solid (as implied in the artist impression), which effects the impact it has on the light curve.

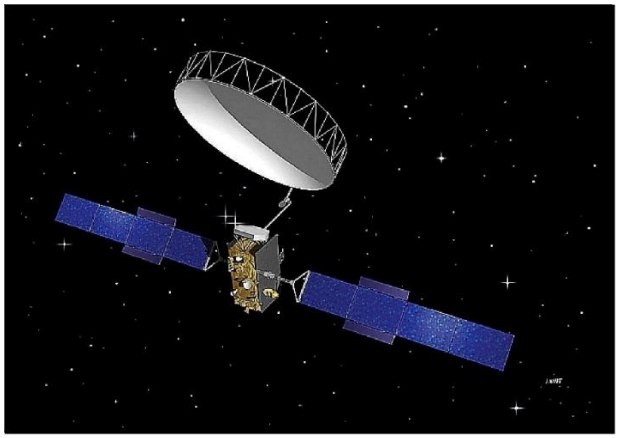


Figure 2. Artist impression of Alphasat in orbit. Image Credit: ESA (https://directory.eoportal.org/web/eoportal/satellite-missions/a/alphasat)

Once the object had been selected, observations were completed using DeSS. A total of 16 nights of observations were performed, though some of these nights were only partial nights due to weather conditions and sensor time constraints. The light curves obtained are shown in Figure 3. Note that the magnitudes of the light curves have been manually altered to separate them in the y-direction, for easier comparison.

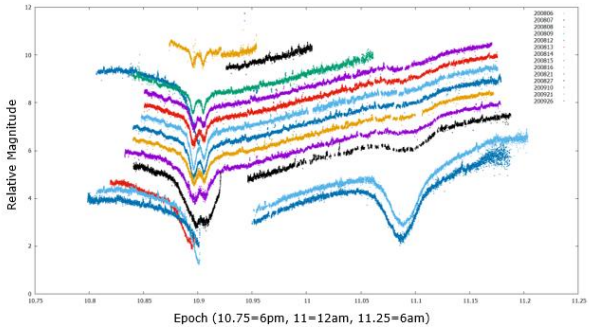


Figure 3. Alphasat light curves gathered by DeSS. Note the light curves have been manually altered to separate them in magnitude for easy comparison.

There are several key features to note in these light curves. First, most obviously, is the consistent 'W' shape appearing in all the curves, this is due to the solar arrays (in particular that the solar arrays are slightly offset from each other, producing a double glint). However, it can be seen that this 'W' shape is not consistent in all curves it varies in depth and width due to the changing lighting conditions from night to night. Also of note is the

appearance of a large dip in the light curve of the later dates, it is believed this is caused by reflection off the side panel of the satellite; the side panels are highly reflective, as such it is believed that the changing orbital geometry of the problem produces this extra dip. Long term study could confirm this supposition.

In the later light curves some large gaps in data can be seen, these are attributable to the shadowing caused by the Earth due to the time of year. This is, unfortunately, unavoidable when using data from a single sensor. It should be noted that the shadowing is gradual, and introduces noise to the curve, obscuring the true magnitude, as can be seen in the lower black curve, prior to the gap in observations. It will be important, for Phase 3, that this type of bias is removed from the curves before the data is used in training the machine learning algorithm.

Less obviously there is a pattern of small, short-lived increases in the noisiness of the curves, which is attributable to the satellite passing across the same background stars, hence increasing the noise. An interesting study would be to attempt to create a filter to remove this systematic noise by analysing the light curves in detail, but that is not in the scope of this project.

In the lowest blue curve, at the end of the observation period, the curve appears to become very noisy. However, this is not a true reflection of the object magnitude, but was induced by mist and worsening sky conditions which eventually meant observations had to be halted. As with the shadowing effect, for Phase 3, it is recommended that such noisy data be removed prior to passing the data to the machine learning algorithm.

Of particular note in these observations is that over a short span of time (less than 2 months), the light curve of a single object alters significantly. This reinforces the conclusion from the Phase 1 study that long term study of the light curves may provide better understanding of the object. It also suggests that a simple machine learning algorithm based on a convolutional neural network (CNN) will not necessarily be able to attribute observations to the correct satellite or even the correct satellite shape due to the possible variance in the light curve. This reinforces a second conclusion from the Phase 1 study: that an algorithm based on a recurrent neural network (RNN) will be necessary to properly characterise space objects. These conclusions are currently being tested.

5 THE LIGHTCURVE SIMULATOR

The light reflected by a space object and measured by a detector located on the Earth is modified by a wide variety of processes. These need to be well-understood in order to separate information from noise in the signal. For most accessible telescope systems, the object under consideration is smaller than the resolved pixel and hence the light collected in that pixel is also affected by materials around the Resident Space Object (RSO) (aka “target”) and stray light within the telescope. The dominant light source is the Sun, but the received light is varied by:

- Object physical motion, especially related to the angle between Sun and observer (Solar Aspect Angle – SAA)
- Physical design and materials
- Motion of panels and antennas on the object
- Emissions from the object
- Background (or foreground) objects, usually stars but could be other debris objects in the pixel field of view
- Errors in the intervening atmosphere, telescope and detection systems

In the analysis of light curves, four main aspects must be considered, the light source, the reflector (or target), the observer and any errors or noise in the data. In most cases, the light source will be the Sun, however, the moon and other light sources can also be considered. The reflector is the target of interest, in most cases a spacecraft or piece of space debris. The observer is the telescope or other technology used to acquire data. Finally, any potential sources or error (noise) in the data must be considered; these include but are not limited to; atmospheric effects, material degradation or even dust on the telescope lens. These four aspects provide the building blocks for the creation of the light curve simulator.

The light curve simulator has been developed in Python3 on an Ubuntu platform. The program POV-Ray (available at: povray.org) was called using python scripts to for the ray-tracing required to produce the light curve.

Python contains a wide library of modules applicable for multiple different purposes. Libraries such as *numpy* and *matplotlib* have been used for basic mathematical functions, while *skyfield*, *spacetrack*, and *astropy* have been used to calculate orbital information for the bodies of interest: the satellite, the Sun, the sensor and the Moon.

POV-Ray is an open-source 3D rendering software which uses ray tracing to generate images. In this context the version used is the one developed for Unix.

Ray tracing is a technique that renders a final image by simulating rays of light travelling in the real world, backwards from the observer to the source. This way the software is able to save a lot of computational time by only simulating the rays that reach the observer. The rays generated from the viewer and pass through the scene. Every time the ray hits an object, the amount of light received by that surface is calculated by sending rays backwards to each light source.

To simulate a scene, POV-Ray reads in a text file containing information written in a specific language

describing the objects and lighting conditions in a scene. Then POV-Ray generates an image of the scene from the point of view of the observer (the telescope in this case), which is also documented in the text file. POV-Ray has an extensive number of features, allowing users to render the scene, both in a realistic or an artistic way, depending on the requirement.

The architecture of the simulator is divided into 6 main steps:

1. Setup
2. Position Propagation and Visibility Analysis
3. Object Creation
4. Image Generation
5. Image Calibration and Processing
6. Data Extraction and Storage

The setup step reads in the required simulation inputs, and determines the mode of simulation; consecutive image creation for a single object, concurrent image creation for a single object, concurrent simulation for multiple objects or batch simulations of multiple objects. This allows a user to determine the processing power drawn and affects the simulation run-time; with batch simulations being the fastest way to generate large volumes of light curve data.

The position propagation and visibility analysis step draws on JPL ephemeris data to determine the positions of the celestial bodies of interest (Sun, Earth and Moon) and on either TLE or OEM ephemeris data for satellite position. Using this data, the orbital geometry of the problem is determined and visibility checks determine if the space object of interest would be in view of the sensor and under the appropriate conditions (i.e., no Earth shadowing, no Moon interference, Sun below horizon etc.). This check dictates at which time steps images of the object could be captured by the real sensor, and hence which should be simulated.

Next, the object is created in the POV-Ray language using constructive geometry; a process of combining simple shapes such as boxes and spheres to create more complex shapes. An object is defined using statements such as union, intersection and difference, allowing a user to combine or subtract shapes from each other as necessary. This can complicate matters significantly, as to create a simple dish geometry for an antenna requires building a sphere then subtracting material using a smaller sphere and a box. However, it has been possible to create every geometry required with a little imagination. For the initial study of the machine learning algorithm, some simple shapes are studied to provide a baseline understanding of what the algorithm will be capable of. For Phase 3 more complex geometries will be implemented, including the complex geometry of Alphasat, used in the validation phase of the simulator. Once the object geometry is defined, the position, attitude and material information is applied.

Once the object is defined and the orbit propagated, the next step in the simulator is to generate images as seen from the simulated sensor. There were significant challenges in this part of the simulator development, POV-Ray imposes limits on the simulation; distances that photons travel and photon-pixel limits. The first of these meant the simulation had to be scaled to allow the program to compute properly the number of photons reaching the sensor from the object. The second, however, was a hard limit that could not be overcome easily. Eventually, a workaround was found by artificially magnifying the image then calibrating it to restore the proper proportion. To create the image, POV-Ray requires the input of the objects included in the simulation, in this case the Object, Sensor, Earth, Sun, and Moon.

After POV-Ray has generated the image, the image is processed. Simply put, the FITS image is ingested and the pixel value for each pixel in the image is evaluated and the values for all lit pixels are summed to find the total brightness of the object. Using real data from DeSS an approximate noise value is applied to the total brightness at this point, using a random distribution to simulate various real random noise sources such as background stars, weather, and atmospheric effects. After processing is complete and noise has been added, the image must be properly calibrated to remove the scaling imposed in the image generation step and translate the POV-Ray output to a real photon count for each pixel of the image.

Finally, after the image processing is complete the outputs are saved as csv files, from which necessary data can later be extracted. The csv files can be parsed and light curves plotted and fed to the machine learning algorithm.

With the simulator complete the first data sets used to train and test the machine learning algorithm could be generated. In the first round of simulations four different types of studies have been conducted; the effects of Shape, Size, Attitude and Materials on the ML classification ability.

6 MACHINE LEARNING

Machine learning techniques, especially usage of deep learning models, have now been researched and implemented in various fields of interest from health problem diagnosis [6] to sound classification [9]. Of particular interest is the state-of-the-art that uses one dimensional data, such as time-series or sequence data as inputs and develop well-performing classification models with accuracies of 70% [1, 3] and few reaching above 90% [1, 4, 9].

The aim for the machine learning part of this project is to develop algorithms that can extract different features of a GEO object from its light curve, in order to identify

various characteristics such as its shape, construction materials, dimensions and subtle differences to the object over time due to the impact from the space environment. The current workflow uses simulated and real data together for model development, however can be adjusted depending on the future work.

For the initial model architecture, an autoencoder (AE) model that independently extracts patterns and features from light curves was selected. Later, a convolutional neural network (CNN) model imports the encoder part of the pre-trained AE as the base model and adds additional CNN layers on top to obtain classification of satellite characteristics. The next stage is to implement a recurrent neural network (RNN) model that will also potentially use the encoder part of the AE as a base model and build RNN layers on top to extract and classify satellite characteristics.

6.1 Autoencoder Development

The autoencoder (AE) is an unsupervised learning model that is used for data compression and pattern extraction from self-learning rather than human engineered labels. In Figure 4, the general structure of the autoencoder is shown. It consists of an encoder part that ingests the input data, extracts features and stores these in a compressed form.

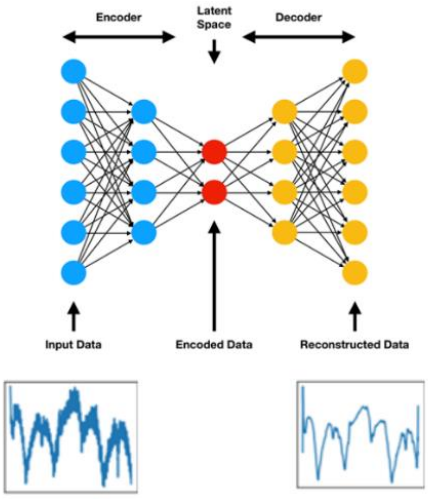


Figure 4. Representation of an Autoencoder model. Adapted from: www.compthree.com/blog/autoencoder

The second part of the autoencoder is the decoder, which ingests the compressed data and attempts to reconstruct the input to its original form. A convolutional autoencoder (CAE) consists of convolutional layers that create dense feature maps that consist of compressed extracted features and patterns that the model has defined as an important pattern of the input data. Convolutional layers are broadly used in image classification models, however, there have also been cases [5, 6, 7] where they have been applied to one-dimensional data in unsupervised learning and reached satisfactory results.

Initially with limited labelled data, multiple cases of different convolutional autoencoders (CAE) were tested to establish the most efficient model that could reconstruct the 1D data input to its original form. Model structures varied from four to six convolutional layers followed by max-pooling layer in encoder model and up-sampling layer in the decoder model. The filter size for each layer augmented by two to increase the number of learnable parameters, however the kernel size, pooling size and the activation function stayed the same for all CAEs. In order to have a better understanding of how well the model is learning and performing, a masking function has been implemented on the loss function and the metrics (except for accuracy) that sets all the predicted values after each epoch to zero in the same indexes where the actual values were missing or padded with the zero value.

From these preliminary trials, the CAE model ('CAE1') architecture was defined with five 1D convolutional layers in both encoder and decoder models and the filter size was set for each layer in corresponding order - 8, 16, 32, 64, 128 for encoder and a reverse sequence for the decoder model. The kernel size was set to three for all convolutional layers. The activation function for all the top layers was set to be the 'ReLu' function, except for the output layer, where a sigmoid function has been selected. In the encoder, a 1D max-pooling layer has been added, with a pooling size of two, to compress the output after each convolutional layer and create a denser feature map of the input. For the decoder, a 1D up-sampling layer is added after each convolutional layer, with a pool size of two, to have the opposite effect, creating a broader feature map of the inputs, hence reconstructing the data to their original form. The 'Adam' optimizer and Mean Square Error (MSE) loss function are used to train the model.

6.1.1 CNN Development

The development of the CNN model was completed in multiple stages. The first; testing was used to determine if a CNN model can make a binary classification between two shapes – sphere and box. If the light curve sample was a box with solar panels then it was labelled as a 'Box'. This model will be referred to as 'CNN1' in this paper. Its architecture was defined by importing the pretrained encoder model and set as non-trainable. This would be the base model on which one additional convolutional layer with filter size of sixteen and kernel size of three was added, with a max-pooling layer following it. The activation function was set as 'ReLu' in the additional convolutional layer and a sigmoid function for the output layer, in order to provide binary classification (either 0 or 1). The 'Adam' optimizer and binary cross entropy are used to train the model.

With more generated cases of simulated data, the next stage was to determine how well the model can classify between three possible satellite shapes – sphere, box and box with solar panels. For further testing, there were two model algorithms tested – 'CNN2' that consisted of a single feature extractor and 'CNN3' that has two parallel feature extractors (see in Figure 5). Both of the models' top layers are set with the 'ReLu' activation function, however for the output layer, the activation function is set as softmax function, in order to provide multiclass probability output ranging between 0 and 1.

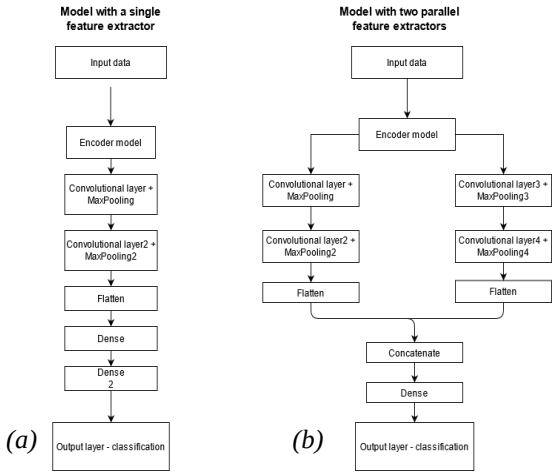


Figure 5. Representation of the two model structures with (a) 'CNN2' having a single, and (b) 'CNN3' having a parallel feature extractor.

For further improvement, a more complex model algorithm defined in [1, 4] has been constructed. The proposed architecture consists of seven layers from which three are convolutional layers with filter size 64, 32, and 64 respectively. After each convolutional layer a dropout and a max-pooling layer is added with the dropout rate being 0.2 and pooling size of 4. This model architecture was adapted into the CNN development for this project, as these papers have done testing on similar data types and their findings showed satisfactory results with an accuracy reaching above 90% when testing the model on simulated data and 70% accuracy on real data. All the papers made use of a similar model architecture with the layer parameters adapted and tested to each case individually.

The 'CNN4' model structure consists of the above-mentioned architecture as the top layers on the non-trainable encoder model with the activation functions set the same as in previous model testing stages. This multiclassification model achieved better results in accuracy and the f1-score compared to previous model cases.

The following stage is the development of a multiclassification model that can classify satellites by its shape and size. The initial training case was conducted using the 'CAE1' model and 'CNN2' model architecture, however with more simulated cases being generated, there have been multiple trials with different model complexity, such as using 'CNN4' model instead of 'CNN2', and testing of different classification cases.

6.1.2 RNN Development

The last model architecture under development is an RNN model that differs from the above two by having memory units or also called cells that contain information generated from previous inputs, and feedback information from previous layers. From the former research papers [2, 8, 10], the two main RNN models under investigation are the Long Short-Term Memory (LSTM) and the Gated Recurrent Unit (GRU).

The first stage for this model development is using an LSTM network for classification of the satellite's shape. The model structure consists of the pretrained encoder model, masking layer and two additional LSTM layers on top. This architecture is a simple model, further referred to as 'LSTM1', and is still under development, hence there has been much testing and comparison of the model's results by changing the LSTM layer unit parameters, as well as trialling the LSTM model without importing the encoder model. The development of both the RNN models is still in the initial testing phase; as more training data is required for these complex models, and Phase 3 will include further model testing and algorithm refinement.

6.2 Data Used for ML

For testing and training multiple different deep learning models with limited data, all the datasets have been split to training, validation and testing sets with 70%, 15% and 15% split for each set. All models train on the same training set and validate after each run on the validation set. The test set is composed of data samples, which none of the models have been trained or validated with, and can be used to assess model performances independently.

6.2.1 Simulated Data

Simulated data has been generated for 4 different case groups - shape, size, attitude and material. In each case one key parameter has been changed in the configuration file (light curve simulator input file) containing the object's properties. The initial cases consist of light curves from one night each month for 12 months and generated for 3 different orbits (based on existing GEO objects).

For the shape cases, there are simulated light curves for three shapes: a sphere, box and box with wings (or box with solar panels). All shapes have a consistent size. These parameters are the base characteristics for all simulated cases except for the size case, where these parameters vary. For the attitude case, object's orientation has been varied. In the size case, the height of

the box is varied ranging from extra small to large size. Similarly, for the sphere shape satellite, the radius of the sphere is varied. For the box with wings case, the box stays consistent, however the solar panel length changes. The small label is applied to all the other simulation cases, as the default size is in the range between the extra small and medium cases. In the material cases, the object configuration currently consists of 3 possible materials for box and sphere shapes.

6.2.2 Real Data

This dataset was available from the start of Phase 2 and consists of real light curves from 16 observational nights of the same satellite, ALPHASAT, as discussed previously. The satellite is classified as a box with wings (though its shape is more complicated than the simple shapes used in the initial simulation cases). Each light curve covers a portion or a full-night of observations, hence has inconsistencies in the data set with varying lengths and frequencies due to effects such as adverse weather conditions or Earth shadowing.

6.2.3 External Data

The Mini-MegaTORTORA (MMT) system is a novel multi-purpose wide-field monitoring instrument built for and owned by the Kazan Federal University, who collaborate and are under agreement with the Special Astrophysical Observatory in Russia.

The set up for the MMT includes nine individual channels installed in pairs on five equatorial mounts. Each channel is equipped with an Andor Neo sCMOS detector with field of view reaching roughly 9x11 degrees with angular resolution of roughly 16" per pixel. The detector is able to operate with a minimum exposure time of 0.03s. Most of the data collected is with 0.1s frequency, hence it provides 10 frames of inputs per second. Most of the data are short length light curves ranging from 1 minute to 1-2 hours.

For this project, the data (available at: mmt.favor2.info) was useful for the first testing of different convolution autoencoder models as there was a limited amount of data to use for model implementation at the start of Phase 2. However, due to the short lengths of the light curves this data set is less ideal for use in this study. Full night light curves are preferable to provide the maximum amount of data on the object of interest.

6.3 Data pre-processing

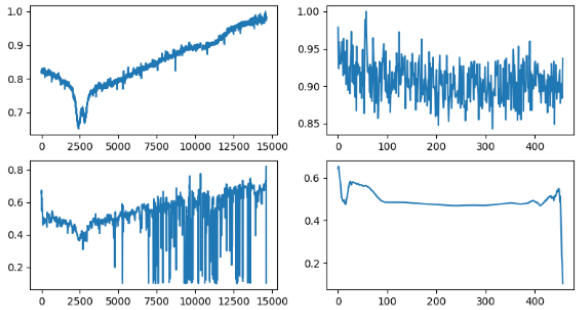
One of the stages in pre-processing for 1D time-series data is data resampling. In order for the 1D data to have a uniform frequency, the light curve is checked to make sure it has a consistent interval between the timestamps, if true it continues with next method of the pre-processing stage. If false, the light curve is resampled to a consistent frequency. Note the frequency is selected to match the frequency of the real data from DeSS. The magnitude value for the resampled timestamp is filled first with the corresponding value from the original sequence. If there isn't a recorded timestamp that matches to the new time, the magnitude value is filled with a zero value that represents the missing values.

Light curves are also scaled to a range from 0 to 1, where the value of 0 is being used for substitution of missing values and data sequence padding to uniform length. This is implemented to bring all the values into the range required for the model to extract patterns from data. In this method, the padding of the data sequence is added, hence if a light curve is shorter than the predefined maximum length, it is padded with zeros at the beginning or the end of the sequence. If the data sample is longer than the defined length, it can be truncated from the either ends of the sequence to match the maximum length. In this model testing, the data was pre-processed with the padding and truncation set to be applied at the end of the data sequence.

6.4 Initial Results

6.4.1 Autoencoder results

The initial results with the MMT data and limited number of DeSS full-night light curves, the graphs showed that it either keeps predicting some noise or the substituted zero values as part of the light curve in both the MMT data and the Deimos real satellite data. The models had difficulty reconstructing the shorter period light curves from the external source as the small variations in magnitude were not predicted. In Figure 6a and Figure 6b is a sample of the results predicted from the initial testing of the 'CAE1' model. It can be observed that the magnitude values of the reconstructed light curve are by 0.2-0.3 units lower than the original light curve.



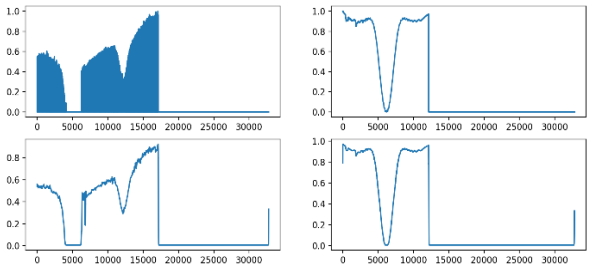
(a) *(b)*

Figure 6. Representation of the results from first 'CAE1' model testing, where (a) displays a sample of Deimos real light curve data and (b) is a sample of MMT satellite light curve. For all of the graphs the 0 values are masked. Top row is with the original pre-processed light curve graphs and bottom row has the predicted light curve plots.

However, with more light curve data generated that

covers full-night observations, the prediction graphs from the 'CAE1' become more precise, with better results. Figure 7 displays the training results for the convolutional autoencoder, where the model is only trained on the simulated data and the few cases of the real observational light curves. The graphs show that the magnitude values of the reconstructed light curves have improved and are predicted within close range to the original value, however there are still drops or peaks at the beginning and end of the light curve. As well as, it can be seen that the 'CAE1' model has learned the small variations in magnitude throughout the night as it can be observed in both of the reconstructed graphs.

Both of the real sample light curves in Figure 6a (top) and Figure 7a (top) have been pre-processed, however in Figure 6a (top) the 0 values have been masked, whereas in Figure 7a (top) they are not.



(a) *(b)*

Figure 7. Representation of the results from an updated 'CAE1' model testing, where (a) shows Deimos real satellite light curve sample and (b) is Deimos simulated light curve sample. The top row is with the original pre-processed light curve graphs and the bottom row has the predicted light curve plots.

Additional note: the samples in these plots are two different light curves from two observational nights. With every new simulation case being produced, the autoencoder is retrained with the new samples.

6.4.2 CNN results

For this stage of CNN model development, the binary classification between the two shapes had good results with accuracy and f1-score both reaching 100% seen in Figure 8a. These results were obtained when all the simulation and real cases were used to pretrain the 'CAE1' model and train the 'CNN1' algorithm. In order to test, how the model was going to cope with a new concept of varied sized satellites, 14 new light curves from the size simulation case were added to the test set. As a result, the accuracy and f1-score of the model decreased to 81% and 85% seen in Figure 8b. This shows that there are differences in the light curves from different size satellites, especially for sphere shapes, that the model has difficulty classifying without being trained on first. It is interesting to note that the algorithm did not ever classify a box as a sphere, only the other way round, it is assumed that this is due to the magnitude range of the light curves being quite small for the original sphere cases, whereas the larger sphere cases have more variation more like the box cases.

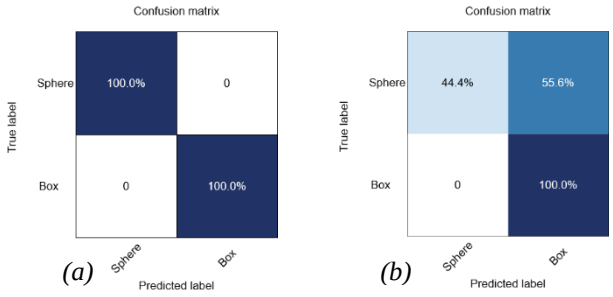


Figure 8. Comparison of the binary classification results on the test set. In (a) the 'CAE1' and the 'CNN1' model has been trained and then tested on the shape and attitude simulation cases. (b) shows the confusion matrix when new samples of the size simulation case have been added only to the test set without retraining the CNN model on the new dataset.

In the multiclassification model, where the CNN model is predicting between 3 different shapes – sphere, box and box with wings, the first results in Figure 9 are satisfactory and can be seen that the 'CNN2' model can distinguish the light curves for sphere and box shapes with the accuracy 90% and above. However, it has difficulty classifying light curve for 'BoxWithWings' shape, with only 38.5% accuracy. The model predicts with 84% accuracy that the object's main structure is a box, however it has difficulty predicting that this structure has solar panels, hence 46% of the samples were classified as a 'Box' class instead of the actual label 'BoxWithWings'.

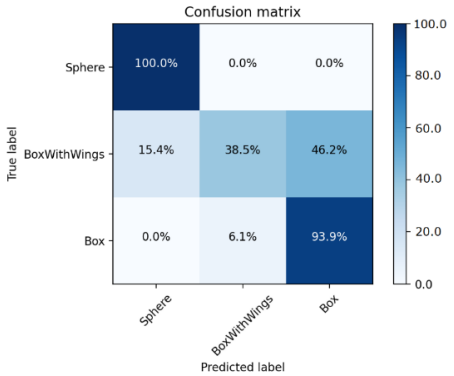


Figure 9. Representation of the preliminary classification results on the test set from 'CNN2' model training.

In further testing, by adding more light curve samples to the training set for each classification case and using the 'CNN4' model, which has more learnable parameters,

the results increased in accuracy and the prediction are more precise as seen in Figure 10. The accuracy for each classification shape is now above 90%. The 6.1% of cases misclassified are 2 attitude case samples that were predicted to be 'BoxWithWings', but the true label is 'Box' class. Further analysing those light curve graphs from that simulation case, it can be seen that these samples have similar features with light curves from the 'BoxWithWings' class in the same orbit. Hence, the model has difficulty discerning the subtle difference. It is expected that by increasing the training data set further these misclassifications could be removed entirely. However, alternatively using an RNN may also be able to improve the results.

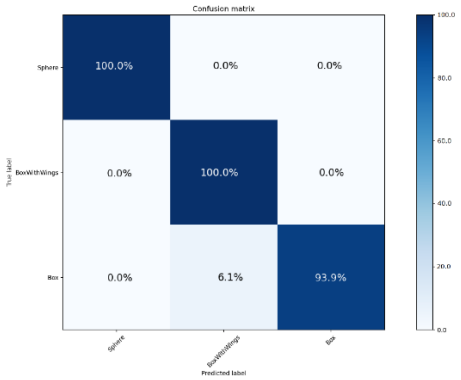


Figure 10. Representation of the classification results on the test set from a 'CNN4' model training.

For the classification case seen in Figure 11 and Figure 12, the 'CNN2' and 'CNN4' model are classifying the light curves on the object shape and size. The results show that with the limited amount of data, it is difficult to distinguish between the different sizes, however it can still efficiently determine light curves for each shape class. Both models are trained on the same data sets, however the 'CNN4' model results have improved in accuracy for sphere shapes improving by 12.5%, for box shape by 3% and for box with wings by 18%. The aim in further model testing would be for all of the values in the diagonal to increase and the other fields in the matrix to decrease.

For further analysis, we trained CNN models on each shape separately to classify the object size. These results showed that the 'Small' label has the best prediction rate with accuracy ranging from 66% for 'BoxWithWings' to 90% for 'Sphere' shape. This is due to the data imbalance between the size cases and also the shape cases (see Table 1), which has an impact on the model performance.

In addition, if the total amount of training data used in the different stages of model development is considered, and compared to the reference papers [1, 3, 4] (where the standard size of training datasets for deep learning models is at least 1000 samples), it can be concluded that the models need more data to produce more accurate classification results. Nevertheless, the findings analysed show that there is good potential in classifying different satellite characteristics.

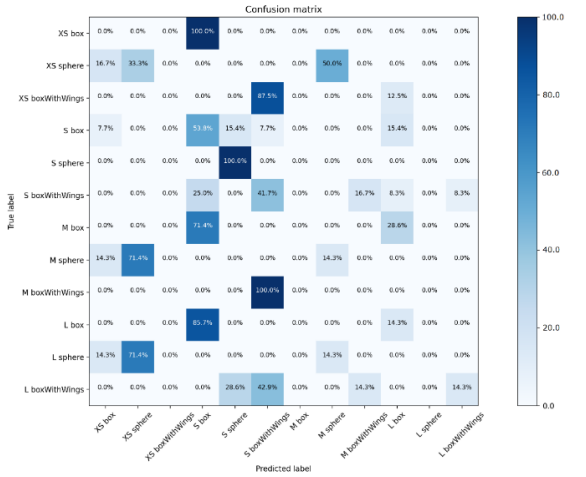


Figure 11. Confusion matrix of the multiclassification on the test set from model testing and training using the 'CNN2' model architecture.

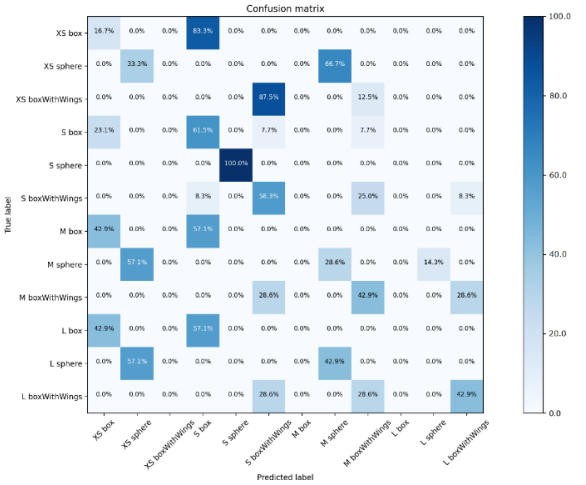


Figure 12. Confusion matrix of the classification results on the test set from further model testing and training using the 'CNN4' model architecture.

Table 1. Number of training samples per category.

Training Set	Box	Sphere	Box with Wings	Total
XS	24	24	22	70
S	62	44	49	155
M	22	22	22	66
L	22	22	22	66
Total	130	112	115	**357**

6.4.3 RNN results

For the first LSTM model training and testing, we developed a multiclassification model predicting object shape. Comparing the results in Figure 12 with the

'CNN4' model predictions in Figure 10, it can be seen that the 'LSTM1' model has more difficulty of detecting the differences in light curves for each shape as this is a more complex model, therefore typically more training data is required. Even though the performance of this model can be improved, the overall accuracy is good with the value reaching above 80%. With more training samples and testing cases, the LSTM model can be expected to surpass these preliminary results in the next project phase.

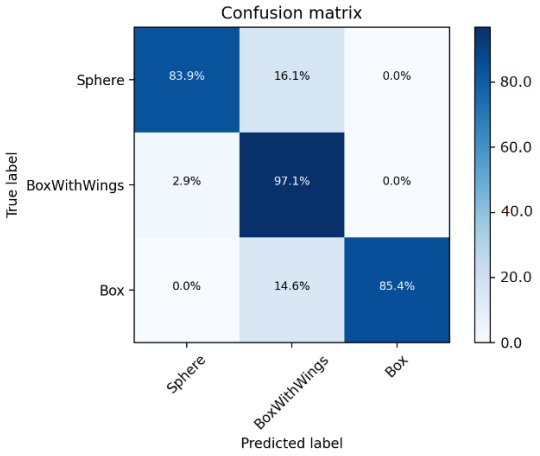


Figure 13. Confusion matrix of the 'LSTM1' model classification results on the test set.

7 CONCLUSIONS AND NEXT STEPS

The light curve simulator has been successfully developed and tested. The first simulation cases have been produced and used in training the machine learning algorithm. Initial testing of the machine learning algorithm has been positive, proving that it will be capable of classifying objects based solely on their light curves.

For Phase 3 of the project, the plan is to refine and improve the developed CAE and CNN models with more training, fine-tuning the layer parameters and testing further classification possibilities on other satellite characteristics. Further test cases proposed include a trial run with the same 3 different models for each shape separately, with the CAE also only be trained on those specific shapes. Another main aspect that will be focused on is the development of the RNN with more testing and training of different possible algorithms for the LSTM and GRU models. An interesting model architecture is presented in [10], where the LSTM layers are trained in parallel to the convolutional layers and in the final layers merged together to output a classification. This model architecture would also be interesting to apply in this project and analyse the results, adapting the layer parameters to the available datasets and project aims.

With the extended dataset provided from DeSS, the simulator and ML algorithms will be tested more extensively.

8 ACKNOWLEDGEMENT

Deimos would like to thank the UK Defence Science & Technology Laboratory (Dstl) for funding this project.

9 REFERENCES

1. Furfaro, R., Linares, R., & Reddy, V. (2018). Space Objects Classification via Light-Curve Measurements: Deep Convolutional Neural Networks and Model-based Transfer Learning. 17
2. Zhang, R., & Zou, Q. (2018). Time Series Prediction and Anomaly Detection of Light Curve Using LSTM Neural Network. Journal of Physics: Conference Series, 1061, 012012. https://doi.org/10.1088/1742-6596/1061/1/012012
3. Hinners, T.A., Tat, K. and Thorp, R., 2018. Machine learning techniques for stellar light curve classification. The Astronomical Journal, 156(1), p.7
4. Furfaro, Roberto, et al. Shape Identification of Space Objects via Light Curve Inversion Using Deep Learning Models. Sept. 2019, p. 17.
5. Z. Chen, C. K. Yeo, B. S. Lee and C. T. Lau, "Autoencoder-based network anomaly detection," 2018 Wireless Telecommunications Symposium (WTS), Phoenix, AZ, USA, 2018, pp. 1-5, doi: 10.1109/WTS.2018.8363930.
6. Abdelhameed, A.M., Daoud, H.G. and Bayoumi, M., 2018, October. Epileptic seizure detection using deep convolutional autoencoder. In 2018 IEEE International Workshop on Signal Processing Systems (SiPS) (pp. 223-228). IEEE.
7. Chen, S., Yu, J. and Wang, S., 2020. One-dimensional convolutional auto-encoder-based feature learning for fault diagnosis of multivariate processes. Journal of Process Control, 87, pp.54-67.
8. Che, Z., Purushotham, S., Cho, K. et al. Recurrent Neural Networks for Multivariate Time Series with Missing Values. Sci Rep 8, 6085 (2018). https://doi.org/10.1038/s41598-018-24271-9
9. Abdoli, S., Cardinal, P. and Koerich, A.L., 2019. End-to-end environmental sound classification using a 1D convolutional neural network. Expert Systems with Applications, 136, pp.252-263.
10. F. Karim, S. Majumdar, H. Darabi and S. Chen, "LSTM Fully Convolutional Networks for Time Series Classification," in IEEE Access, vol. 6, pp. 1662-1669, 2018, doi: 10.1109/ACCESS.2017.2779939