General-sum Game Modeling of Generative Adversarial Networks for Satellite Maneuver Detection

Dan Shen[a], Carolyn Sheaff[b], Genshe Chen[*a], Mengqing Guo[a], Erik Blasch[c], and Khanh Pham[d]

[a] *Intelligent Fusion Technology, Inc., 20271 Goldenrod Lane, Germantown, MD USA 20876;*
[b] *Information Directorate, Air Force Research Laboratory (AFRL), Rome, NY USA 13441;*
[c] *Air Force Office of Scientific Research (AFOSR), Arlington, VA USA 22203;*
[d] *Space Vehicles Directorate, Air Force Research Laboratory (AFRL), Kirtland Air Force Base, NM USA*

ABSTRACT

Space protection and SSA require rapid and accurate space object behavioral and operational intent discovery. The problem of behaviorally evasive intent identification is challenging and complicated. The satellite maneuver detection and classification is the first step of space behavior discovery. With exiting capabilities based on anomality detection, classifying orbits as normal or abnormal ones is lacking maneuver details to support space object behavioral and operational intent discovery. We enabled the GANs for SDA by improving the training data efficiency via modeling the conflicts, between discriminator and generator, under a general-sum game, rather than a zero-sum game structure. General-sum game, enables the "persistence of excitation", by adding small noises in the loss functions. To solve the general-sum game during the training, we use a Fictitious-play framework, under which the discriminator outputs converge to the optimum discriminator function, and the mixed output from the sequence of trained generators converges to the data distribution. We test our GAN models on the scenario of observing LEO satellites from ground-based radars. The input data include azimuth angle, elevation angle, range, range rate, principal RCS and orthogonal RCS. We use small set of data for training and obtained promising detection and classification results (94.5% accuracy, 9.74 in inception score and 0.002 in Frechet inception distance) on the evaluation data. These performance metrics show the proposed GANs for SDA can automatically, accurately and rapidly detect and classify the satellite maneuvers to support the further space behavioral and operational intent discovery.

Keywords: Space domain awareness, GANs, satellite characterization, sensor models, machine learning, general-sum games.

1. INTRODUCTION

With recent world-wide developments, society has increasingly relied on space assets to provide sensing and communications benefits in various industrial, civilian, and commercial applications. Since information from space is critical for various decisions, space is considered an advanced frontier. In addition to real-time and hidden information constrains, the existence of space object density significantly produces the complexity of the space situational awareness (SSA). For accurate SSA, there are multiple challenges such as (*i*) partially observable movements, (*ii*) resident space objects (RSOs), (*iii*) uncertainties modeling and propagation, (*iv*) real-time respondence, and (*v*) computationally intractable algorithms.

The consideration of space access investigation and mission trade-offs is critical for space-borne operations. Tracking algorithms that can measure space objects based on collected data to track satellites, debris, and natural phenomena (such as comets, asteroids, and solar flares). Understanding the position of space objects from low-level information fusion can support high-level information fusion SSA missions of sensor, user, and task refinement [1]. In order to implement SSA accurately, it is possible to coordinate the evaluation of residential space objects (RSO) through user-defined operation pictures (UDOP) [2]. Improvements to tracking and sensor SSA include models (such as track

*gchen@intfusiontech.com; phone 1 301515-7261; fax 1 301 515-7262

mechanics), measurement, calculation software (such as tracking), and application-based system coordination (such as situations). For example, the *game theory* method used in SSA can be used for pursuit and escape analysis [3].

There are several papers recently applying deep learning methods for space situation awareness in the area of space object detection, classification, and identification [4][5]. The National Imagery Interpretability Rating Scale (NIIRS) and video NIIRS (V-NIIRS) can be utilized to determine the dynamic behavior analysis of objects [6][7], where the V-NIIR is modified from the traditional NIIRS [8]. However, due to the data link bottleneck restrictions, large images are collected and transmitted from air or space domain to ground domain needs compression, which changes the NIIRS ratings [9-12]. Therefore, a brokering system was developed to determine the image collection parameters based on the NIIRS (or V-NIIRS) requirements [13][14]. Similar system is such like Light Detection and Ranging (LIDAR [15]) and Thermal infrared imagery [16]. Other works, such as works by Lucas, Kyono et al [17][18], has developed to utilize deep learning neural networks to analysis Space-Object NIIRS (S-NIIRS), where a convolutional neural networks was utilized to improve the classification resolution (smaller ground resolved distance due to enhanced imaging characteristics for space telescopes) of imagery from SNIIRS.

Deep learning, convolutional neural networks, as well as other machine learning tools really help the development of the space image processing, object tracking, and information fusion technologies [19][20]. In addition, some advanced deep learning networks, such as generative adversarial networks (GAN) can provide capability for generating fake data for the same distribution of the input samples. Therefore, SSA can be also developed by using GAN model to create more circumstances such as jamming [21], collision avoidance [22], and coordinated collection [23] for robustness analysis. The enhanced classification neural networks [24][25] will improve the performance of SSA at the same time.

Space superiority requires space protection and space domain awareness (SDA), which rely on rapid and accurate space object behavioral and operational intent discovery. The satellite maneuver detection and classification is the first step of space behavior discovery. With exiting capabilities based on anomality detection, classifying orbits as normal or abnormal ones is lacking maneuver details to support space object behavioral and operational intent discovery. Consequently, an automatic maneuver detector with enough details for SDA in the detection results is desired.

Machine learning and artificial intelligence (ML/AI) is a good candidate for satellite maneuver detection and classification. The generative adversarial networks (GANs) are data-hungry and rely heavily on vast quantities of diverse and high-quality training examples in order to generate accurate satellite maneuver detector. However, unlike image processing, the SDA has limited training data available. Moreover, GANs remain remarkably difficult to train and existing approaches to improve GAN's train data efficiency still rely on heuristics that are extremely sensitive to modifications.

In this paper, we modify the game setup of generators and discriminators in GANs. The zero-sum game in existing GANs is replaced by a general sum game with objective functions containing small Gaussian disturbances. We will train GANs with a Fictitious play concept framework, within which each player (generator or discriminator) presumes that the opponents (discriminator or generator) are playing stationary (possibly mixed) strategies. At each round, each player thus best responds to the empirical frequency of play of their opponent. The GAN contains two adaptive control problems. In the adaptive control of generator, the generator tries to obtain the discriminator's control policy then optimize his own objective function. Same strategy works for the discriminator. From the perspective of adaptive controls, these small Gaussian noises in the cost function can help satisfying the persistence of excitation, which guarantees convergence without a priori stability assumptions and ensures robustness properties. Thus, the proposed modification improves the training data efficiency and enables the application of GANs in the SDA.

The modified GANs are applied in the satellite maneuver detection and classification from the ground-based sensing data, which includes the Azimuth angle (rad), elevation angle (rad), range (km), range rate (km/s), principal RCS (m^2) and orthogonal RCS(m^2) of space objects. The realistic sensor data (azimuth angle, elevation angle, range, range rate) are propagated using SGP4/SDP4 and the various maneuver strategies from the proposed space game model, which is played by on-ground radar and space objectives. We use two-player Markov game to investigate the sensor management for tacking evasive space objects. The Markov game to investigate the sensor management for tracking evasive space objects. The Markov game approach provides a method to solve SSA behavior detection problems, where the Resident Space Objects (RSO) will exploit the sensing and tracking model to confuse the SA observer by corrupting their tracking estimates, while SA observer wants to improve the tracking performance. The different cost functions will generate different maneuver strategies.

We simulate the low Earth orbits (LEOs) based on the two-line elements (TLEs) from space-track.org and the simplified general perturbation version 4 (SGP4) propagator. A small set of 100 orbits and 100-second simulated tracks with various maneuvers (different pointing angles and magnitudes, and maneuvering periods) are used for training the proposed GANs. We used a resampling and interpolation to normalize the input data to the GANs. The sampling frequency of the input data is 0.08 Hz, ie, the gap between the sensing data is 12.4 seconds.

The modified GANs demonstrate the training convergence on the small set of training data. We achieved 94.5% accuracy on the evaluation data. To evaluate the proposed GANs for SDA, we also compute several quantitative measures, such as average log-likelihood, inception score (IS), Frechet Inception Distance (FID), maximum mean discrepancy. We obtained 9.74 in IS and 0.002 in FID.

This paper is organized as the following. A pursuit-evasion game model is detailed in Section 2 to simulate various satellite behaviors. Section 3 introduces the traditional GANs. Section 4 presents our proposed general-sum games enabled GANs for space behavior discovery. The numerical results and analysis are displayed in Section 5. Finally, Finally, Section 6 summarize the whole paper with conclusion.

2. PURSUIT-EVASION GAME MODEL

2.1 System States and Dynamics for Space Objects

To perform maneuvers, the evader (space object being tracked) will apply the continuous low-thrust such as such as the Ion thrust. Ion thrusters tend to produce low thrust, which results in low acceleration. For example, a NASA Solar Technology Application Readiness (NSTAR) thruster producing a thrust (force) of 92 mN will accelerate a satellite with a mass of 1,000 kg by 0.092 N / 1,000 kg = 0.000092 m/s2. The magnitude of the thrust is assumed to be fixed and small. The controls of the thrust commands are the directional angles of these thrusts.

The following equations describe the kinematics and dynamics of the spacecraft's states with continuous low-thrust:

$$\dot{r} = v \sin\gamma \tag{1}$$

$$\dot{v} = \frac{T}{m}\cos\alpha\cos\beta - \frac{\mu\sin\gamma}{r^2} \tag{2}$$

$$\dot{\gamma} = \frac{v\cos\gamma}{r} + \frac{T}{m}\frac{\sin\alpha\cos\beta}{v} - \frac{\mu\sin\gamma}{r^2 v} \tag{3}$$

$$\dot{\xi} = \frac{v\cos\gamma\cos\zeta}{r\cos\phi} \tag{4}$$

$$\dot{\phi} = \frac{v\cos\gamma\sin\zeta}{r} \tag{5}$$

$$\dot{\zeta} = \frac{T}{m}\frac{\sin\beta}{v\cos\gamma} - \frac{v\cos\gamma\sin\phi\cos\zeta}{r\cos\phi} \tag{6}$$

There are six system states: r (norm of position vector), v (norm of velocity), γ (angle between velocity vector and the local horizon plane), ζ(angle between local east and the projection of velocity vector in local horizon plane), ξ (longitude) and ϕ (latitude). α, β are the directional angles of the thrust T.

Given a local direction (α, β) and a Δt, the system states are propagated by following the steps:

1. Convert the states to the local coordinate system
2. Using numerical integration method to compute the local states at $t_0 + \Delta t$
3. Convert the new local states to the ECI

The conversion between Earth-centered inertial (ECI) and local coordinate system (East-North-Up, or ENU) is detailed in the following equations, where (dx, dy, dz) is a vector in ECI and (de, dn, du) is a vector in ENU coordinate system.

$$\begin{bmatrix} de \\ dn \\ du \end{bmatrix} = \begin{bmatrix} -\sin\xi & \cos\xi & 0 \\ -\sin\phi\cos\xi & -\sin\phi\sin\xi & \cos\phi \\ \cos\phi\cos\xi & \cos\phi\sin\xi & \sin\phi \end{bmatrix} \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} \tag{7}$$

$$\begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} = \begin{bmatrix} -\sin\xi & -\sin\phi\cos\xi & \cos\phi\cos\xi \\ \cos\xi & -\sin\phi\sin\xi & \cos\phi\sin\xi \\ 0 & \cos\phi & \sin\phi \end{bmatrix} \begin{bmatrix} de \\ dn \\ du \end{bmatrix} \tag{8}$$

2.2 Pursuit-Evasion Game Setup

In our dynamic PE game model, each player P (purser) or E (evader) has its own system states and state transitions:

$$x_p = [r_p, v_p, \gamma_p, \xi_p, \phi_p, \zeta_p]^\top = [x_1, x_2, x_3, x_4, x_5, x_6]^\top \tag{9}$$

$$x_e = [r_e, v_e, \gamma_e, \xi_e, \phi_e, \zeta_e]^\top = [x_7, x_8, x_9, x_{10}, x_{11}, x_{12}]^\top \tag{10}$$

$$u_p \in \{0, 1\} \tag{11}$$

$$u_e = [\alpha_e, \beta_e]^\top \tag{12}$$

$$\dot{x}_p = g_p(x_p) \tag{13}$$

$$\dot{x}_e = g_e(x_e) \tag{14}$$

Eq. (9-10) and Eq. (13-14) are the augmented versions of Eq. (1-6) for pursuer and evader. Since the purser (a specific ground station) will not maneuver, $T_p = \alpha_p = \beta_p = 0$. The u_p is an on-off control. When $u_p = 1$, the pursuer will spend the sensor resource to get measurements of the evader. When $u_p = 0$, the pursuer will not observe the evader. The controls of the pursuer will not affect the position and velocity states, but it will change the tracking performance in terms of entropy of the covariance, *P,* matrices in the Extended Kalman Filter (EKF) or Cubature Kalman Filter (CKF). With the covariance information, the entropy is:

$$h(x) = -\int \dots \int_{-\infty}^{\infty} f(x) \ln f(x) ds_1 \dots ds_6 \; = \; \frac{1}{2} ln((2\pi e)^6 det\,(P)) \; \propto \; \lambda_1\lambda_2\lambda_3\lambda_4\lambda_5\lambda_6 \tag{15}$$

where $f(x)$ is the probability density function (pdf) of satellite states x (positions and velocities), $s_1, s_2, \dots s_6$ are the components of x, λ_1, $\lambda_2 \dots \lambda_6$ which are the eigenvalues of P matrix. Intuitively, the entropy is proportional to the product of eigenvalues of P matrix. In our PE game setup, the evader will try to increase the entropy while evader wants it to be minimal. The entropy becomes the confliction parameter between the evader and the pursuer.

The cost function of evader is defined in eq. 16, where $\bar{x}_e$ is the filtered state of evader and P is the covariance matrix of a tracker (EKF or CKF).

$$\begin{aligned} J_e = h(\bar{x}_e) &= -\int_{-\infty}^{\infty} \dots \int_{-\infty}^{\infty} f(\bar{x}_e) \ln f(\bar{x}_e) \mathrm{d}\bar{x}_7 \dots \mathrm{d}\bar{x}_{12} \\ &= \frac{1}{2} \ln\left((2\pi e)^6 \det(P)\right) \propto \prod_{i=6} \lambda_i(P) \end{aligned} \tag{16}$$

The purser's cost function is defined in eq. 17, where $P_{\bar{m}}$ and P_m is the error covariance matrices of a tracker without measures, and with measures, respectively. $c > 0$ is a design parameter of fixed reward for saving sensor resources.

$$J_p = \begin{cases} \prod_{i=6} \lambda_i(P_{\bar{m}}) - \prod_{i=6} \lambda_i(P_m) & \text{if } u_p = 1; \\ c & \text{if } u_p = 0. \end{cases} \tag{17}$$

Then the optimal controls of two game players are $u_e^* = \arg\max_{u_e} J_e$ and $u_p^* = \arg\min_{u_p} J_p$. We can solve the game problem using a numerical method [26][27] as well as the Fictious Play concept [28][29].

3. GENERATIVE ADVERSARIAL NETWORK MODEL

A Generative Adversarial Model is a model that is only given the input variables (x) and the problem does not have any output variables (y). A statistical model is constructed by extracting or summarizing the patterns of the input data. In supervised learning, data is labeled to develop a model to predict a class label given an example of input variables. An example of the predictive modeling task is called classification, which is also traditionally referred to as discriminative model. Alternatively, an unsupervised model summarizes the distribution of input variables that may be also to be used to create or generate new examples in the input distribution. As such, these types of models are referred to as generative models. Therefore, a really good generative model may be able to create new examples that are not just plausible, but indistinguishable from real examples from the problem domain.

Generative Adversarial Networks, or GANs, are a deep-learning-based generative model. More generally, GANs provide an architecture for training a generative model, and it is most common to use deep learning models in a GAN architecture. The GAN architecture was first described Ian Goodfellow [30]. The GAN model architecture involves two sub-models: a *generator model* for generating new examples and a *discriminator model* for classifying whether generated examples are real (from the domain) or fake (generated by the generator model). The generator is a model that is used to produce new plausible examples from the problem domain. The discriminator is a model that is used to classify examples as real vs. fake, true vs. imposter, or noise vs. implausible. Generative adversarial networks are based on a game theoretic scenario in which the generator network must compete against an adversary. The generator network directly produces samples. Its adversary, the discriminator network, attempts to distinguish between samples drawn from the training data and samples drawn from the generator.

A generator model takes a fixed-length random vector as input and generates a sample in the domain, such as an image or other data types. A vector is typically drawn randomly from a Gaussian distribution and is used to seed or source of noise for the generative process. It has no meaning other than the meaning applied by the generator model. After training, points in the multidimensional vector space will correspond to points in the problem domain, forming a compressed representation of the data distribution. This vector space is referred to as a *latent space*. A latent space provides a compression or high-level concepts of the observed raw data such as the input data distribution. In the case of GAN, the generator model applies meaning to points in a chosen latent space, such that new points drawn from the latent space can be provided to the generator model as input and used to generate new and different output examples. An easy scheme of generator is shown in Figure 1.

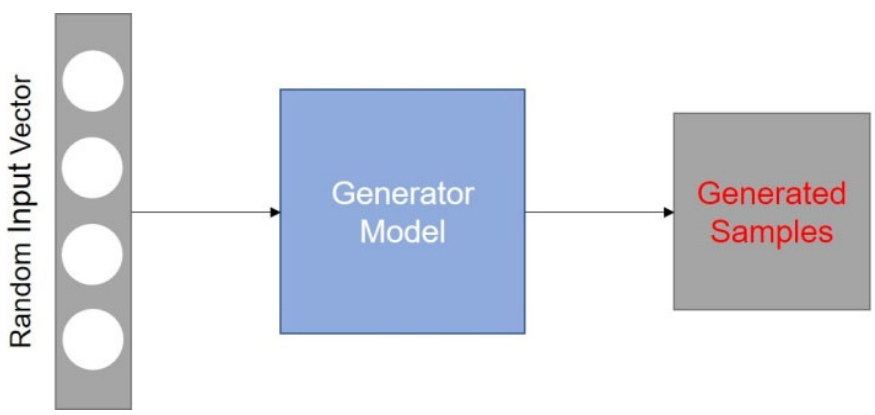

Figure 1. Example scheme of the GAN Generator Model.

On the other hand, the discriminator model takes an example from the problem domain as input (real or generated) and predicts a binary class label of real or fake (generated). The real example comes from the training dataset. The generated examples are output by the generator model. The discriminator is a normal classification model. After training process, the discriminator model is discarded. However, discriminator model is very important that one way to improve the performance of GAN is to create a stronger discriminator model. An example scheme of the GAN discriminator model is represented as Figure 2.

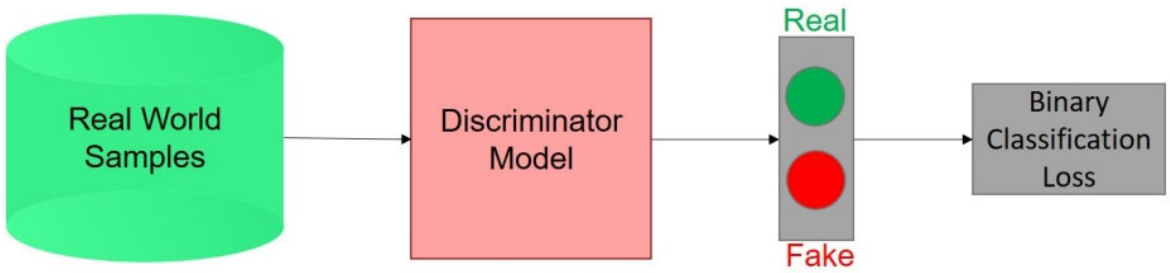

Figure 2. Example scheme of the GAN Discriminator Model.

The two models, the generator and discriminator, are trained together for GAN architecture. As mentioned earlier, the generator generates a batch of samples, and these, along with real examples from the domain, are provided to the discriminator and classified as real or fake. Then, the discriminator is updated to get better model, and importantly, the generator is updated based on how well, or not, the generated samples fooled the discriminator. In this way, the two models are competing against each other. They are adversarial in the game theory and are playing a zero-sum game. Note that "zero-sum" means that when the discriminator successfully identifies real and fake samples, it is rewarded and no change is needed to the model parameters, whereas the generator is penalized with large updates to model parameters. On the contrary, when the generator fools the discriminator, it is rewarded and no change is needed to the model parameters, but the discriminator is penalized, and its model parameters are updated. At a limit, the generator generates perfect replicas from the input domain every time, and discriminator cannot tell the difference and predicts unsure (50%) every case. The GAN architecture is shown as Figure 3.

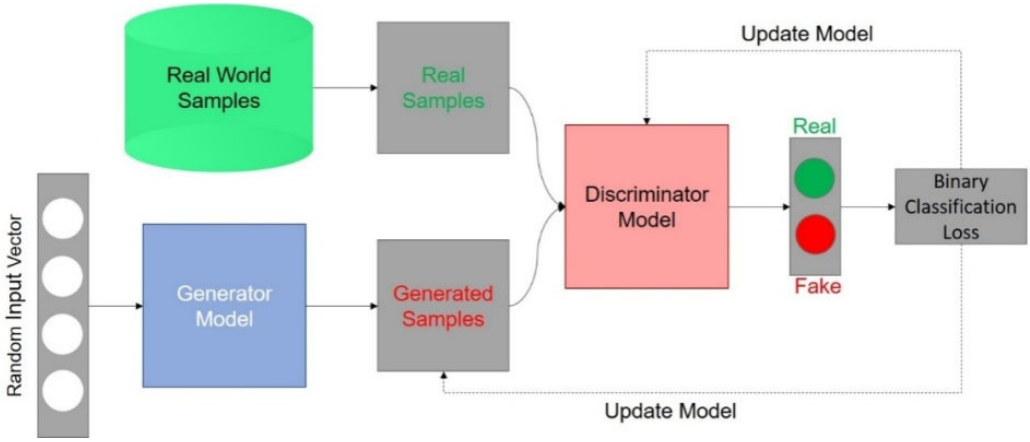

Figure 3. Scheme of the Generative Adversarial Network Model Architecture.

4. GENERAL-SUM GAME ENABLED GANS

Generally, GANS are data-hungry and rely heavily on vast quantities of diverse and high-quality training examples in order to generate accurate satellite maneuver detector. However, unlike image processing, the SDA has limited training data available. Moreover, GANs remain remarkably difficult to train and existing approaches to improve GAN's train data efficiency still rely on heuristics that are extremely sensitive to modifications. This section describes methods to exploit the persistence of excitation [31] as well as general-sum games [32]-[40] to improve the training performance of GANs. The purpose of deep learning is to discover rich, hierarchical models that represent probability distributions over the kinds of data encountered in artificial intelligence applications. To enhance SSA, the conditional GAN (cGAN) is used in the RSO behavior classification. GAN models can perform semi-supervised learning reasonably well. Meanwhile, a GAN model can generate more data from the input data to improve the robustness of the trained model to increase the test accuracy of input data.

4.1 Conditional GAN Models for Space Behavior Detection

Instead of a traditional GAN model, it is desirable to make use of the class-lable information to train our GAN model to generate useful satellite tracks. Therefore, the GAN can be improved by class label to prohibit the mode collapse. Likewise, based on the data quality that is available from the SNIIRS, there is a priority factor in the GAN training. Another motivation is to generate targeted satellite tracks. The way of using the labeled (or scored) information is called a *Conditional Generative Adversarial Network* (cGAN) [41]. By using the cGAN model, it is possible to direct the data generation process. The approach is utilizing one hot encoded with class label and concatenating with the input to both the generator and discriminator models. As shown in Figure 4, the scheme of conditional generator and conditional discriminator in a Conditional Generative Adversarial Network is displayed. There are many advantages of using conditional GAN models. First, the configuration of cGAN models produces a reliable result in the stable training of GAN model for a wide variety of problems. A best practice involves using an embedding layer followed by a fully connected layer with a linear activation that scales the embedding to the size of the image before concatenating it in the model as an additional channel or feature map.

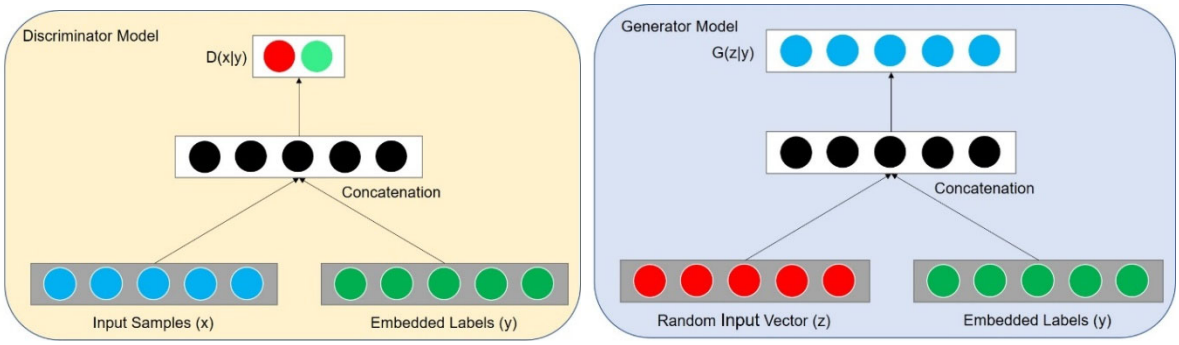

Figure 4. Scheme of a Conditional Generator and a Conditional Discriminator in a Conditional Generative Adversarial Network.

4.2 General-sum Game Enabled GANs

As shown in Figure 3, the GAN model architecture connects Generator (G) as well as Discriminator (D). G and D plays a zero-sum game. Each player (G, or D) tries to adaptively learn the opponent. Similar to adaptive control problems, the successful adaptiveness depends on the satisfaction of the persistence of excitation [31] during the interactions. A simple way to meet the persistence of excitation is to include some small white noises in the cost functions of player. This results a general-sum game instead of a zero-sum game between D and G of GANs. Hence the general-sum game enabled cGAN is modified as follows:

a) The input to the generator network is $x_g = [z, l]$, which is concatenate version of a random latent noise vector $\mathbf{z} \in \mathbb{R}^{\boldsymbol{d}}$sampled from $\mathcal{N}(0,1)$ and a one hot encoding of the class label, $l \in \{0,1\}^{(N_c)}$with N_c real classes.

b) The discriminator mapping D takes the real data x or the generated image $G(x_g)$ with the data's label as input and output one distributions real/fake, which is the probability of the input being real. It is modeled as binary classifier.

The proposed iterative optimization procedure is summarized as a pseudocode in Algorithm 1.

Algorithm 1: Iterative Training Procedure of Enhanced GANs

Minibatch stochastic gradient descent training of generative adversarial nets. The batch size to apply to the discriminator, k, is a hyperparameter.

1: **training iterations = N**

2: **for t in 1:N do**

3: Sample k (batch_size) samples with labels from real dataset $\mathcal{R}$: $\{r_i, y_i\}_{i=1}^{k}$

4: Sample k random latent noise samples $\{z_i\}_{i=1}^{k} \sim \mathcal{N}(0,1)$

5: Let z_i and assigned labels $l_j (j \in (0,1,2,\ldots,10))$ be the concatenated inputs to generator

6: Let $f_i = G(z_i, l_j)$ be the generated datasets for fake data

7: Update the discriminator using the following objectives:

$$L_D = L_{real} + L_{fake} + +\alpha\mathcal{N}(0,1) \tag{18}$$

$$L_{real} = max_D \frac{1}{k}\sum_{i=1}^{k} \log\big(D(r_i, y_i)\big) \tag{19}$$

$$L_{fake} = max_D \frac{1}{k}\sum_{i=1}^{k} \log\Big(1 - D\big(G(z_i, l_j), l_i\big)\Big) \tag{20}$$

where α is a weighting distribution parameter.

8: Update the generator, only for the fake data, through the discriminator gradients

$$L_G = min_D \frac{1}{k}\sum_{i=1}^{k} \log\Big(1 - D_{real}\big(G(z_i, l_j), l_i\big)\Big) + \alpha\mathcal{N}(0,1) \tag{21}$$

9: **end for**

The gradient-based updates can use any standard gradient-based learning rule.

In details, the algorithm is shown as below.

1. Given real orbital data as input, d_k (k = 1, …n) outputs one distributions D. D is optimized by minimizing a binary cross entropy loss L_D. In the case of real inputs, the gradients are generated using the following loss function:

$$L_D = \mathbf{E}_{r\sim\mathcal{R}} \max_D \log\big(D(r)\big) + \log\Big(1 - D\Big(G\big(r_{fake}\big)\Big)\Big) + \alpha\mathcal{N}(0,1) \tag{22}$$

2. Using the gradients from D, G is updated using the adversarial loss to produce realistic class consistent real orbital data.

$$L_G = \min_G \mathbf{E}_{z\sim Z} \log\Big(1 - D\Big(G\big(r_{fake}\big)\Big)\Big) + \alpha\mathcal{N}(0,1) \tag{23}$$

where α $(0 < \alpha < 1)$ is the percentage of the loss function value, and the $\mathcal{N}(0,1)$ is the Gaussian noise with 0 mean and 1 standard deviation for the noise data.

To enhance the game theoretic training enabled deep learning (GTEL) SSA methods, a Space unveiled Behavior GAN (SuB-GAN) is proposed. The SuB-GAN generator and discriminator were constructed based on the dimensions of the input orbital data. The whole model built based on the 10-label classification model as well as the conditional GAN model. As shown in Figure 5**,** the architecture of generator networks is based on the conditional GAN model. The input of the generator contains two parts, which are latent noise dimension as well as input label. The latent noise contains 200 dimensions with Gaussian noise with 0 mean and standard deviation of 1. And the input label contains 10 different numbers for generating different satellite orbits. Both two parts bypasses embedding layers or dense layers to increase its complexity before concatenating together to a 3*3*257 tensor. Then this tensor will pass several Conv2Dtranspose layers to the 3*15*1 tensor as the generated orbit results, which have similar domain as the source or target inputs. In the main Convolutional layer part in the GAN model, instead of first four layers, the convolutional layers mostly have 2*2 filters and follow another two simple design rules: (*i*) the first four layers only have 1*x dimensions to upsample our tensor from 3*3 dimension to 3*15 tensor dimensions, and (*ii*) the filter sizes has increased first and then decreases in the 2 times order except for the last layers. Since GAN model is better not to have Maxpooling layer as well as Stride in the generator or discriminator, we perform the upsampling as well downsampling directly by filter size without "same" padding settings. The total number of weighted convolutional layers is 8. The details of all the parameters of generator are shown in Table 1**.**

As shown in Figure 6, the Discriminator Networks is based on the 10 label classification model. The input of discriminator is 3*15*1 tensor dimensions with the real label of the data. The input bypass includes 3 convolutional layers as well as 2 dense layers with 30% and 15% dropout layers. In order to predict the 10 labels classifications as well as the fake or real sample, there are only one set of outputs, which is a 1 degree sigmoid output for binary classification. The output comes from the last dense layer with 64 neurons. In addition, the Sub-GAN uses the ADAM optimizer for our models during the training process with adjustable learning rate as well. The details of all the parameters of D-Net are shown in Table 2. The total parameters are listed in Table 3.

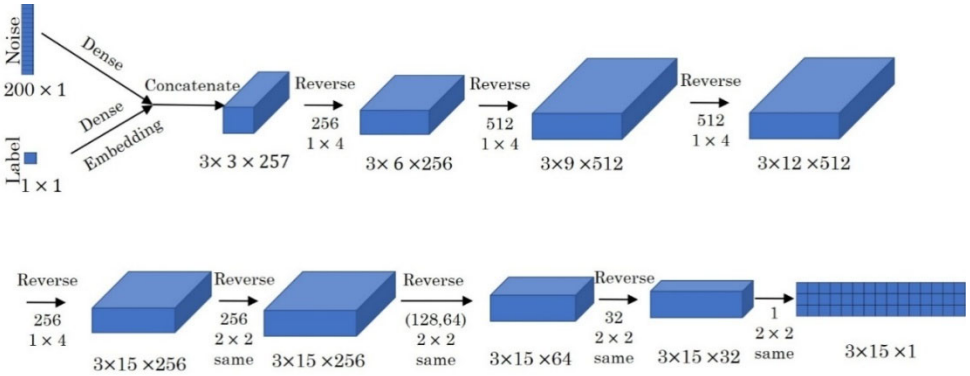

Figure 5. The Architecture of Generator Networks.

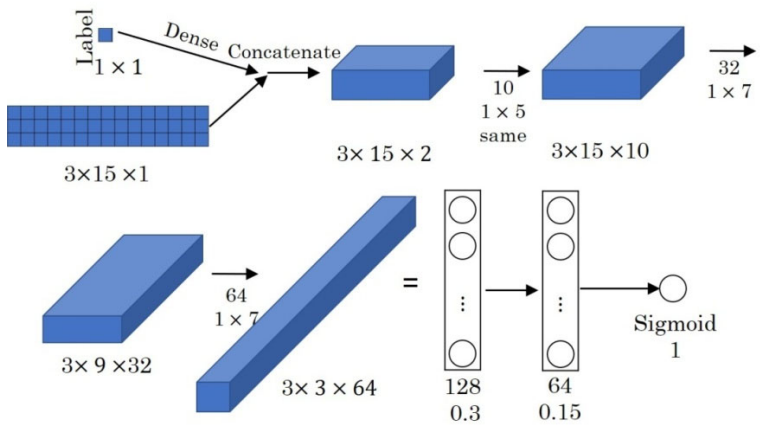

Figure 6. The Architecture of Discriminator Networks.

Table 1 The parameters of our conditional GAN system for generator

Layer(type)	**Output Shape**	**Param #**
Embedding	(None, 1, 200)	2000
Dense	(None, 384)	77184
Dense	(None, 1, 256)	51456
Dense	(None, 768)	295680
Dense	(None, 1, 128)	32896
Dense	(None, 2304)	1771776
Dense	(None, 1, 9)	1161
Conv2d_Transpose	(None, 3, 6, 256)	263424

Conv2d_Transpose	(None, 3, 9, 512)	524800
Conv2d_Transpose	(None, 3, 12, 512)	1049088
Conv2d_Transpose	(None, 3, 15, 256)	524544
Conv2d_Transpose	(None, 3, 15, 256)	262400
Conv2d_Transpose	(None, 3, 15, 128)	131200
Conv2d_Transpose	(None, 3, 15, 64)	32832
Conv2d_Transpose	(None, 3, 15, 32)	8224
Conv2d	(None, 3, 15, 1)	129

Table 2 The parameters of our conditional GAN system for discriminator

Layer(type)	**Output Shape**	**Param #**
Embedding	(None, 1, 200)	2000
Dense	(None, 1, 45)	9045
Conv2d	(None, 3, 15, 10)	110
Conv2d	(None, 3, 9, 32)	2272
Conv2d	(None, 3, 3, 64)	14400
Flatten	(None, 576)	0
Dense	(None, 128)	73856
Dropout	(None, 128)	0
Dense	(None, 64)	8256
Dropout	(None, 64)	0
Dense	(None, 1)	65

Table 3 Parameters of ResNet GAN for 143 labels

Name	**ResNet Generator**	**Plain Discriminator**
Generator Parameters	Trainable Parameters	Non-Trainable Parameters
5,028,794	5,028,794	0
Discriminator Parameters	Trainable Parameters	Non-Trainable Parameters
220,008	110,004	110,004
GAN Parameters	Trainable Parameters	Non-Trainable Parameters
5,138,798	5,028,794	110,004

5. NUMERICAL RESULTS

5.1 Training Data Generation

In order to generate the training data, the tracks based on the TLEs from space-track.org were modified by adding behavioral maneuvers from section 2. The different maneuvers indicate different labels in our training data. As an example, the maneuver to increase the orbital energy in azimuth angle by 15 degrees is labeled as 1. On the other

hand, an increase in the orbital energy in both azimuth and elevation angles for 15 degree is labeled as 13. The details for maneuver addition are depicted as follows:

a. The earth-centered inertial coordinates (ECI) are computed from the two-line element (TLE) at 0 time-step;
b. Use the Markov Game methods to propagate the satellites tracks;
c. Convert the 16 waypoints back to azimuth angle, elevation angle, range, range rate relative to a ground site.

Fig. 7 displays the first 10,000 tracks that are from the 72,000 generated tracks for training purposes and another 6,700 tracks for testing.

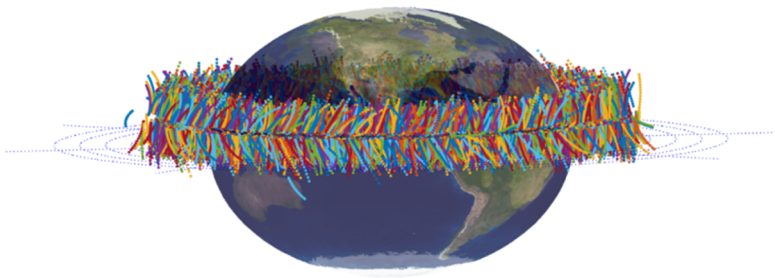

Fig. 7: The First 10,000 Training Tracks with Various Space Behaviors

The data format is listed as (Each row is an observation):

- Column 1: track id
- Column 2: observation id (from 1 to 15)
- Column 3: Azimuth angle (rad)
- Column 4: Elevation angle (rad)
- Column 5: Range (km)
- Column 6: Training label (from 1 to *m*, where *m* is total types of space behaviors)

5.2 Numerical Results and Discussions

The generated data provides inputs to the previously trained classification model to determine the classification accuracy and determine the accuracy of the the generator model. The accuracy results of our generator model with noise in the cost function are shown in Figure 8. With more than 3500 epochs of training for our both generator and discriminator model, the SuB-GAN system achieved around 93% accuracy of the generator, which produce 93% accuracy data from the previous classification model. It indicates the great performance of our trained generator used in the analysis.

In addition, the generated data was plotted the confusion matrix as shown in Figure 8 (right side), where only the last two labelsare not accurate. However, most of the labels have an accuracy higher than 90% for every labels. Therefore, both results show a tremendous performance for our SuB-GAN model with noise in cost function, which leads to stabilized training performance for our modified conditional GAN model.

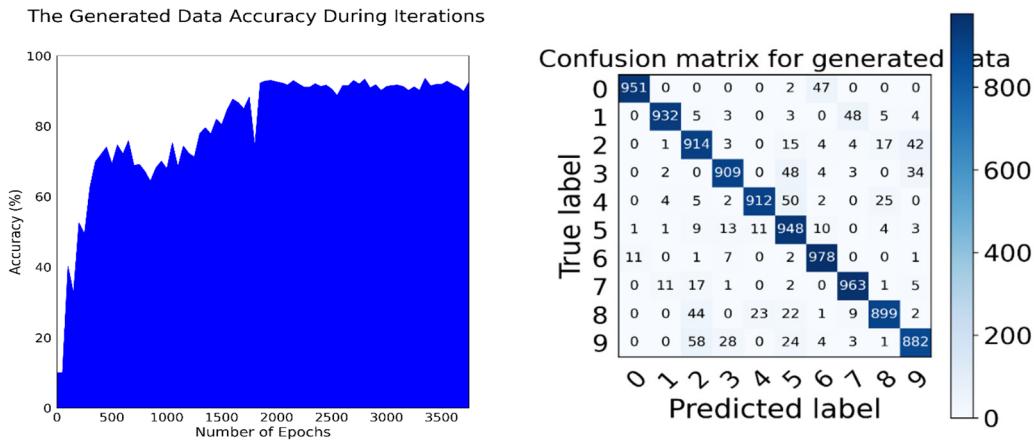

Figure 8. The Results of our trained GAN model.

Other than the results above, there are several quantitative measures to evaluate our GAN generator model, such as average log-likelihood, inception score (IS), frechet inception distance (FID), maximum mean discrepancy, and etc. Here, the inception score and frechet inception distance are presented to evaluate the SuB-GAN model due to the beneficial for these two measures for conditional GAN model.

The *inception score* is an objective metric for evaluating the quality of generated data, specifically synthetic images output by generative adversarial network models. The inception score involves using a pre-trained deep learning neural network model for image classification to classify the generated orbital data. Specifically, the probability of the image belonging to each class is predicted. The inception score mainly calculate the Kullback-Leibler divergence between the conditional and marginal probability distributions. The equation of this divergence is shown below:

$$KL\ divergence = p(y|x) \times (\log(p(y|x)) - \log(p(y))) \tag{24}$$

The KL divergence is then summed over all images and average over all classes and the exponent of the result is calculated to give the final inception score. Thus, the inception score has a lowest value of 1.0 and a highest value of the number of classes supported by the classification model. In the experiments, inception score is ~9.7, which is very close to the number of classes, 10. Therefore, the trained generator can produce a great distribution near our original orbital data.

Another quantitative measure is called *Frechet Inception Distance score*, or FID for short. FID is a metric that calculates the distance between feature vectors calculated for real and generated images. The score summarizes how similar the two groups are in terms of statistics on features of the raw samples. Lower FID scores indicate the two groups of the data are more similar, or have more similar statistics, with a perfect score being 0.0 indicating that the two groups of images are identical.

The FID score is calculated by first loading a pre-trained classifier, the output of the model is removed and the output is taken as the activations from the last pooling layer, a global spatial pooling layer. Then the feature vector is then predicted for a collection of real images from the problem domain to provide a reference for how real the data is represented. Feature vectors can then be calculated for synthetic data. The FID score is then calculated using the following equation:

$$d^2 = \left|\left|mu_1 - mu_2\right|\right| + Tr(C_1 + C_2 - 2 \times \sqrt{C_1 \times C_2}) \tag{25}$$

where mu_1 and mu_2 refer to the feature-wise mean of the real and generated data. The C_1 and C_2 are the covariance matrix for the real and generated feature vectors. Our trained generator got 0.002 FID score, indicating the fabulous performance our GAN training process with noise in the cost function.

Note, the use of the IS and FID are similar to the opportunities from the NIIRS, VNIIRS, and SNIIRS. Thus future efforts will exploit these relationships for tracking the SuB-GAN to meet operational considerations for user and machine processing.

6. CONCLUSIONS

The goal of the paper describes the deep learning method to determine space object behaviors. The paper describes an adaptive control method that provides additional small noises for the persistence of excitation, to enhance a traditional conditional GAN. It utilizes a Fictitious Play concept to solve the GANs with general-sum games. The numerical results show that adding noise in the cost function of the model improves the stability during the GAN training process. Results demonstrate an effectively trained generator model to produce synthetic data of space object orbital data. The generator produces a 93% accuracy of classification model as well as the 9.7 Inception score and 0.002 Frechet Inception Distance (FID) score. It indicates the outstanding performance of the enhanced GAN model. Future work will explore the feasibility of relating the small noises to the training loss.

REFERENCES

[1] Blasch, E., Bosse, E., Lambert, D. A. *High-Level Information Fusion Management and Systems Design*, Artech House, Norwood, MA, 2012.

[2] Blasch, E., Enhanced Air Operations Using JView for an Air-Ground Fused Situation Awareness UDOP, *AIAA/IEEE Digital Avionics Systems Conference*, Oct. 2013.

[3] Shen, D., Pham, K, Blasch, E., Chen, H., Chen, G, Pursuit-evasion orbital game for satellite interception and collision avoidance, *Proc. SPIE*, Vol. 8044, 2011.

[4] McQuaid , I., Merkle, L. D., Borghetti, B., Cobb, R., Fletcher, J., Space Object Identification Using Deep Neural Networks, *AMOS*, 2018.

[5] Kyono, T., Lucas, J., Werth, M., Flecther, J., McQuaid, I., Determining Multi-frame Blind Deconvolution Resolvability using Deep Learning, *AMOS* (2019).

[6] Kahler, B., Blasch, E., Predicted Radar/Optical Feature Fusion Gains for Target Identification, *Proc. IEEE Nat. Aerospace Electronics Conf (NAECON)*, 2010.

[7] Blasch, E., Kahler, B., V-NIIRS Fusion Modeling for EO/IR Systems, *IEEE Nat. Aerospace and Electronics Conf.*, 2015.

[8] Blasch, E., Kahler, B., Application of VNIIRS for target tracking, *Proc. SPIE*, Vol. 9473, 2015.

[9] Chen, H-M., Blasch, E., Pham, K., et al., An investigation of image compression on NIIRS rating degradation through automated image analysis, *Proc. SPIE*, Vol. 9838, 2016.

[10] Blasch, E., Chen, H., Wang, Z., et al., Target Broker Compression for Multi-Level Fusion, *IEEE National Aerospace and Electronics Conference*, 2016.

[11] Zheng, Y., Chen, G., Wang, Z., et al., Image quality (IQ) guided multispectral image compression, *Proc. SPIE 9871*, 2016.

[12] Blasch, E., Chen, H-M., Wang, Z., et al., Compression induced image quality degradation in terms of NIIRS, *IEEE Applied Imagery Pattern Recognition Workshop*, 2016.

[13] Chen, H-M., Wang, Z., Chen, G., J. M. Irvine, et al., Tailoring image compression to mission needs: Predicting NIIRS loss due to image compression. *Proc SPIE*, 2018.

[14] Blasch, E., Chen, H-M., Irvine, J. M., et al., Prediction of compression induced image interpretability degradation, *Opt. Eng*. 57(4), 043108 2018.

[15] Duan, Y., Irvine, J. M., Chen, H-M., et al., Feasibility of an Interpretability Metric for LIDAR Data, *Proc SPIE,* 2018.

[16] Chen, H-M., Irvine, J. M., Wang, Z., et al., Predicting Interpretability Loss in the Thermal IR Imagery due to Compression, *IEEE Applied Imagery Pattern Recognition Workshop*, 2018.

[17] Lucas, J., Kyono, T., Werth, M., Flecther, J., McQuaid, I., Automated Resolution Scoring of Ground-Based LEO Observations Using Convolutional Neural Networks, *AMOS*, 2019.

[18] Kyono, T., Lucas, J., Werth, M., Claef, B., McQuaid, I., Flecther, J., Machine learning for quality assessment of ground-based optical images of satellites, *Opt. Eng*. 59(5), 051403 2020.

[19] Jia, B., Pham, K. D., Blasch, E., et al. Cooperative space object tracking using space-based optical sensors via consensus-based filters, *IEEE Transactions on Aerospace and Electronic Systems* 52 (4), 1908-1936, 2016.

[20] Blasch, E., Ravela, S., Aved, A. (eds.), *Handbook of Dynamic Data Driven Applications Systems*, Springer, 2018.

[21] Wei, M., Chen, G., Cruz, J. B., et al., Multi-pursuer multi-evader pursuit-evasion games with jamming confrontation, *Journal of Aerospace Computing, Information, and Communication* 4 (3), 693-706, 2007.

[22] Shen, D., Pham, K., Blasch, E., et al., Pursuit-evasion orbital game for satellite interception and collision avoidance, *Proc. SPIE*, Vol. 8044, 2011.

[23] Jia, B., Blasch, E., Pham, K. D., et al., Space Object Classification using Fused Features of Time Series Data, *Advanced Maui Optical and Space Surveillance Technologies (AMOS)*, 2017.
[24] Shen, D., et al. Adaptive Markov inference game optimization (AMIGO) for rapid Discovery of satellite behaviors. *Sensors and Systems for Space Applications XII*. Vol. 11017. International Society for Optics and Photonics, 2019.
[25] Shen, D., Sheaff, C., Guo, M., Blasch, E., Pham, K., & Chen, G. (2020, May). Enhanced GANs for satellite behavior discovery. *In Sensors and Systems for Space Applications XIII* (Vol. 11422, p. 114220F). International Society for Optics and Photonics.
[26] Shen, Dan, Khanh Pham, and Genshe Chen. "A Numerical Solution to Orbital Pursuit-Evasion Games." *amos* (2018): 61.
[27] Shen, D., et al. Computer vision and pursuit–evasion game theoretical controls for ground robots. *Advances in Mechanical Engineering* 11.8, 1687814019872911, 2019.
[28] Brown, G. W. Iterative Solutions of Games by Fictitious Play, In *Activity Analysis of Production and Allocation*, T. C. Koopmans (Ed.), New York: Wiley, 1951.
[29] Berger, U., Brown's original fictitious play, *Journal of Economic Theory* 135:572–578, 2007.
[30] Goodfellow, I., et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.
[31] Boyd, S. and Sastry, S., "Necessary and sufficient conditions for parameter convergence in adaptive control." *Automatica*, 22.6 629-639, (1986).
[32] Shen, D., et al. "Computer vision and pursuit–evasion game theoretical controls for ground robots." *Advances in Mechanical Engineering* 11.8, 1687814019872911, (2019).
[33] Shen, D., Pham, K., Chen. G., "A Numerical Solution to Orbital Pursuit-Evasion Games," *The Advanced Maui Optical and Space Surveillance Technologies Conference*, (2018).
[34] Pham, K. D., et al. "Methods and devices for demonstrating three-player pursuit-evasion game," *U.S. Patent* No. 9,737,988. 22 Aug. 2017.
[35] Shen, D., et al. "A game theoretic DRA approach for improved spread spectrum frequency hopped waveforms performance in the presence of smart jammers," *Cognitive Communications for Aerospace Applications Workshop (CCAA)*. IEEE, (2017).
[36] Shen, D., et al., "Game optimal sensor management strategies for tracking elusive space objects*." IEEE Aerospace Conference*. (2017).
[37] Shen, D., et al., "Pursuit-evasion game theoretic uncertainty-oriented sensor management for elusive space objects," *IEEE National Aerospace and Electronics Conference (NAECON) and Ohio Innovation Summit (OIS)*. IEEE, (2016).
[38] Shen, D., et al., "Network survivability-oriented Markov games (NSOMG) in wideband satellite communications*," IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC)*. IEEE, (2014).
[39] Chen, G., Shen, D., Kwan, C., Cruz, J., Kruger, M., Blasch, E., "Game Theoretic Approach to Threat Prediction and Situation Awareness", *Journal of Advances in Information Fusion*, 2(1). 35-48, 1 June (2007).
[40] Blasch, E., et al. "Review of game theory applications for situation awareness." *Sensors and Systems for Space Applications VIII*. Vol. 9469. International Society for Optics and Photonics, (2015).
[41] Mirza, M., and Osindero, S., "Conditional generative adversarial nets,"" *arXiv preprint arXiv*:1411.1784 (2014).