Towards operational application of Deep Reinforcement Learning to Earth Observation satellite scheduling

Adrien Hadj-Salah, Jonathan Guerra, Mathieu Picard, Mikaël Capelle

▶ **To cite this version:**

Adrien Hadj-Salah, Jonathan Guerra, Mathieu Picard, Mikaël Capelle. Towards operational application of Deep Reinforcement Learning to Earth Observation satellite scheduling. 2020. hal-02925740

HAL Id: hal-02925740
https://hal.science/hal-02925740

Preprint submitted on 30 Aug 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Towards operational application of Deep Reinforcement Learning to Earth Observation satellite scheduling*

Adrien Hadj-Salah[1,2], Jonathan Guerra[1,2], Mathieu Picard[1,2], and Mikaël Capelle[1]

[1] IRT Saint-Exupéry
[2] Airbus Defence & Space

Abstract. Scheduling an agile optical Earth Observation satellite (AEOS) requires to choose a limited number of images to take among a large set of possibilities. It is a NP-hard problem which is made even more difficult by the presence of uncertainties: an image is useless if it is too cloudy and weather uncertainty cannot be removed, no matter the accuracy of the forecasts. Moreover, among the various types of customer requests of commercial satellites, we focus here on large area acquisitions that cover a country or a continent. Such requests require several months or years to complete even with a constellation of satellites. Considering such long time frames, the completion time highly depends on weather uncertainties and there are currently no trustworthy forecasts. Therefore the selection of the requests is crucial to speed up the completion using a long-term strategy. Reinforcement Learning is an interesting solution to explore when it comes to uncertain environments. We propose to use the well-known *Actor Critic* (A2C) algorithm combined with *Transfer Learning, Domain Knowledge and Domain Randomization* (TDDR). We demonstrate how transfer learning is a way to address real-world problem. We find that TDDR method challenge state-of-the-art heuristics for satellite scheduling on various real weather conditions.

Keywords: Satellite scheduling · Reinforcement Learning · Weather forecast uncertainty · Transfer Learning

1 Introduction

Optical Earth Observation (EO) systems are broadly used to acquire cloud-free images and deliver them to various customers on a daily basis. Acquisition scheduling for EO satellites is a growing topic with the emergence of satellite constellations and the ability to interoperate different systems. The objective of the scheduling algorithms is to maximize the number and quality of acquired images while satisfying various satellite and system constraints.

* Supported by IRT Saint-Exupéry

In this article we focus on the acquisition of large areas, which can typically cover countries or even continents, using Agile Earth Observation Satellites (AEOS). Large area coverages require several months to complete even with multiple satellites. Considering such long time frames, weather uncertainty is one of the major issue. None of the current forecasting systems can reliably predict cloud cover dynamics, especially far in the future. Thus, the problem we are trying to solve is both highly combinatorial but also stochastic.

A review of agile EO satellite scheduling literature over the last 20 years [14] shows that most systems currently rely on heuristic approaches. Heuristics have the advantage of being explainable, have small execution run-times and are compatible with multi-criteria optimization. While exact methods and metaheuristics are also often explored in academic works [5], those methods have two major drawbacks: firstly, exact methods do not scale to realistic use cases. Secondly, these methods are usually not designed to take uncertainties into account. A short review of traditional approaches for satellite constellation scheduling is proposed in Section 2.

In this article, we explore a Deep Reinforcement Learning (DRL) approach to tackle the EO scheduling problem. RL is well tailored to our problem as it provides robust long-term strategies in uncertain environment that outperform traditional short-term heuristics [9]. Besides, it requires no fine-tuning by human-experts and its computation time, after training, is often similar to simple heuristic methods. A review of already existing solutions based on DRL is done in Section 3.

Applying Reinforcement Learning to EO satellite scheduling has already been studied in [6]. The solution proposed is applied on a simplified environment where weather uncertainties are entirely simulated using a normal law and the satellite capacity is limited to one mesh acquired per pass. This work proposes an innovative solution to apply DRL to a higher fidelity environment, detailed in Section 4, based on real weather forecasts and observations, and allowing to acquire more than one mesh during each satellite pass, paving the way towards operational transfer. Our solution, presented in Section 5, uses transfer learning and domain knowledge to build realistic observed weather distributions and domain randomization. The results are presented and discussed in Section 6.

2 Satellite constellation scheduling and traditional approaches

AEOS can perform attitude maneuvers and thus offer multiple opportunities to acquire images depending on their orientation. A constellation of EO satellites is therefore capable of taking hundreds up to thousand images every day. Scheduling for EO satellites consists in selecting acquisitions to be made based on priorities, weather forecasts, system constraints and other mission-dependent criteria. The plan, which consists in a series of attitude maneuvers that the satellite shall execute to take images, is computed from the ground by the Mission Planning Facility (MPF) before the corresponding telecommands are uploaded to the satellite.

Among the various types of imagery requests, we focus on large area coverage requests as represented on Figure 1. Each requests consists in a set of meshes covering the area to acquire. It requires months or even a year to acquire them all. Moreover, uncertainties have a huge impact on the validation of the image acquired. On such long time-frames, weather uncertainty is even more crucial since there are no accurate weather forecasts. The selection of acquisitions according to the weather forecasts directly impacts the overall time-to-completion of the request.

Fig. 1. France completion in progress.

The MPF within the satellite ground segment aims at delivering cloud-free images as fast as possible to attract new customers. In order to be reactive and handle last minute requests, mission planning algorithms are often based on simple heuristics. Greedy algorithms in mission planning are commonly used [8]. Several authors have explored genetic algorithm [8, 11] or Mixed-Integer Linear Programming (MILP) [5]. Such algorithms on real world scenarios take a long time to compute a schedule. Furthermore, they are not intrinsically designed to manage uncertainties which is a crucial point for large area coverage.

Deep Reinforcement Learning has the potential to deal with uncertain environments and to generalize well to new situations [10]. DRL can excel in determining efficient long-term strategies and it has recently achieved breakthrough results on many tasks including complex video-games [13]. The purpose of this paper is to find such a long-term strategy for our large area coverage problem.

3 Background on DRL application to EO satellite scheduling

Application of machine learning to EO scheduling is quite new according to [14]. Papers investigating the benefits of Reinforcement Learning for satellite scheduling are scarce. Besides an application to satellite telecommunications [4], we only know of two papers related to scheduling EO satellites using DRL:

- The application in [6] is similar to our problem but the demonstration is done in a very simplified environment: the weather data for the validation are simulated using a normal law and the satellite capacity is limited to a single mesh acquired per satellite pass over the area of interest.
- Authors in [15] use a two-phase method where a policy selects acquisitions while another policy optimizes the satellite maneuvers for the selected acquisitions. Experiments are done on a public EO dataset (ROADEF'2003) and therefore it contains multiple request types. Their focus is on the exact computation of the mission plan to upload to the satellite. Thus, they do not focus on a long-term strategy tackling weather uncertainties.

In this paper, we focus on learning a policy that selects acquisitions part of a large-area request in a view to minimize the completion duration, taking

into account weather uncertainties. One of the major improvements of this work over [6] is the application of DRL on an environment using real weather forecasts and observations. In order to generalize to unseen data, DRL needs to perform a substantial amount of episodes during learning. Unfortunately, available weather archives are limited, thus the learning process cannot yield a proper policy if the dataset is limited to real weather data. We consequently introduce a new method to simulate weather during learning, while the evaluation is still performed on real archive weather data. We also improve representativity by allowing multiple meshes to be acquired during each pass of a given satellite.

4 Reinforcement Learning formulation

The problem formulation is a generalization of the one described in [6]. We first remind the definition of the problem and then provide its Reinforcement Learning formulation.

4.1 Problem definition

We consider the problem of acquiring a set of meshes using multiple satellites. The meshes are positioned in a regular grid covering the area to acquire, as represented on Figure 1. Some cells of the grid need not be acquired. Each time the satellite passes over the area of interest, it can acquire a subset of the meshes (based on its orbit and agility). The goal of the algorithm is to decide, for each satellite and each pass, which meshes should be acquired in order to minimize the time required to acquire all meshes.

To ease the use of a neural network, the observation space is enclosed in a rectangular box containing the Area Of Interest (AOI) using a Mercator projection. Without loss of generality, we consider that this box is containing $N_{lat} \times N_{lon}$ meshes. Let $\mathcal{M} = \{m_1, \ \ldots, \ m_K\}$ be the set of meshes to acquire with $K \leq N_{lat} \times N_{lon}$ since the AOI is included in this box. Each mesh m is identified by the location of its center, thus we denote $m_k = (lat_k, lon_k)$.

A satellite pass $t \in \{1, \cdots, T\}$ corresponds to a time interval during which a satellite orbit overflies the area to acquire. Because each pass is linked to a given satellite, the accesses and the steps will only be indexed by its corresponding pass. During each pass t, only a subset of the meshes is accessible. Let $\mathcal{M}_t \subseteq \mathcal{M}$ be the subset of meshes in the AOI that can be acquired by the corresponding satellite knowing its orbit and agility.

We assume that the agent can acquire a fixed number of meshes $N_m > 0$ during each pass. This accounts for the agility of the satellite and acquisition durations. This is a generalization of the use case presented in [6] since the MPF can decide to allocate more than one mesh to this large area coverage at each satellite pass. The agent has to choose which of those meshes will be acquired at each pass.

Considering optical remote sensing, a mesh and the associated image is validated only if the observed cloud cover is under a given threshold c_{max}. When the plan is computed, only cloud cover forecasts are available. We denote by $c_t^f(m)$

the forecast cloud cover over mesh m during pass t, and by $c_t^o(m)$ the observed cloud cover. This last value is then used to decide if the acquisition is validated or not. In case of rejection and because each of the meshes has to be validated only once, the mesh remains candidate for future passes.

4.2 Application of Reinforcement learning to large coverage with Earth Observation satellites

The above problem can be formalized as a **Markov Decision Process** (MDP) which is an intuitive and fundamental formulation for Reinforcement Learning (RL). This formulation is inspired by the one defined in [1]. It models a problem composed of an agent interacting with an environment through actions.

A MDP can therefore be defined as a tuple $\langle \mathcal{S}, \mathcal{A}, P, R \rangle$, where the agent is in a state $s_t \in \mathcal{S}$ and takes an action $a_t \in \mathcal{A}$ at each pass $t \in \mathbb{N}$. From the state s_t after taking action a_t, the agent moves to a new state s_{t+1} with a probability of transition $P(s_t, a_t, s_{t+1}) = \mathbb{P}(s_{t+1}|s_t, a_t)$. For each transition, the current state is updated. It takes the value of the next pass date in the chronological order. The list of meshes to acquire is updated: if a mesh m is taken, it is validated if $c_t^f(m) \leq c_{max}$ and still to acquire otherwise. Section 5.2 details the distribution used to validate meshes during learning. At each step a reward $R(s_t, a_t, s_{t+1})$ is given. Since a client cannot wait indefinitely to receive the requested images, we assume the horizon is finite. Therefore, there is a finite number of discrete time steps t during an episode. Each episode comprises a maximum of $T \in \mathbb{N}^*$ steps.

The goal of RL is to find an optimal policy $\pi^\star : \mathcal{S} \to \mathcal{A}$ maximizing the expected discounted reward over this finite horizon:

$$\pi^\star = \arg\max_{\pi} \left\{ \sum_{t=0}^{T} \gamma^t R(s_t, \pi(s_t), s_{t+1}) \right\}$$

where $0 \leq \gamma < 1$ is the discount factor

State space In DRL, this corresponds to the input of the neural network used to derive the policy. In our case, it provides information about the mesh status (whether a mesh has already been acquired or not) and the forecasted cloud cover $c_t^f(m)$ for the next N_{pass} passes. The first pass corresponds to the one for which the agent has to decide which meshes to acquire. The state is thus defined as a tensor of size $(N_{lat},\ N_{lon},\ N_{pass}+1)$ where the third dimension corresponds to the validation frame, following by the N_{pass} forecast frames. More explicitly, the validation frame is a matrix of size $(N_{lat},\ N_{lon})$ where each entry is set to 0 if the mesh has already been validated, and 1 otherwise. The other N_{pass} matrices correspond to the forecasted clear sky fraction extracted from the weather archives for time step t, which is equal to $1 - c_t^f(m)$. A clear sky forecast of 0 is used to indicate that a mesh is not accessible during a given pass.

Action space As mentioned in Section 4.1, N_m meshes have to be selected at each pass t. Thus, each action a_t is a tuple $a_t = (a_t^1,\ \ldots,\ a_t^{N_m})$ where each

$a_t^k \in \{0, \ldots, K\}$ corresponds to a mesh to acquire, with 0 being the do-nothing sub-action (to allow the agent to acquire less than N_m meshes). In addition, it is possible to have several do-nothing sub-actions but it is impossible at a given step to take more than once the same mesh.

Reward To train the agent to acquire meshes that have a higher probability of validation, a reward of 1 is given for each mesh validated during the pass. Therefore:

$$R(s_t, a_t, s_{t+1}) = \sum_{1 \leq k \leq N_m} \mathbb{1}\left(c_t^f(m_{a_t^k}) \leq c_{max}\right)$$

For the sake of simplicity, we consider that $c_t^f(m_0) \leq c_{max}$ is always false.

4.3 Limits to learn an off-the-shelf DRL model on a realistic environment

To be representative of reality, we use ERA-Interim [2] to obtain values for $c_t^f(\bullet)$ and $c_t^o(\bullet)$. To evaluate the DRL method, it is therefore the source used to determine $c_t^f(\bullet)$ and $c_t^o(\bullet)$. ERA-Interim is an archive of global atmospheric reanalysis with a spatial resolution of approximately 80km and a temporal resolution of 6 hours. It provides forecasts from 6 hours ahead to 10 days. It is sufficient for our use-case to build the observation tensor for the neural network and to validate the chosen meshes.

However, learning on an environment picking directly $c_t^f(\bullet)$ and $c_t^o(\bullet)$ in the dataset is not straightforward. Archive weather data available is limited and this causes overfitting when the agent learns on too few different scenarios. Section 5.2 details a way to simulate observed weather from weather forecasts. This allows us to increase the diversity of the learning samples.

Another improvement in the fidelity of the environment compared to [6] is to choose multiple meshes per satellite passes. Nevertheless, the high combinatory drastically increases the action space size. The dimension of the action space directly impacts the number of weights of the policy NN, it is thus important to control it. To get around this and use a similar A2C algorithm as the one proposed in [6], Section 5.3 details a workaround to limit the neural network output size without changing the action space size.

5 Using TDDR for generalization to a realistic environment

TDDR stands for Transfer learning, Domain knowledge and Domain Randomization method [12]. It proposes a modification of the DRL methodology to address the limits identified in Section 4.3.

To overcome the limited number of training samples while learning on real weather data, we propose to learn the model on simulated weather data before evaluating it on real data, as explained in Section 5.1. The generation of weather data is made with a Bernoulli distribution, function of the cloud cover forecast,

and derived from a statistical analysis of a weather archive. To increase diversity of the learning samples, several distributions can be built, using subsets of the weather archive, in order to sample differently observed cloud cover during learning. We detail in Section 5.2 a way to build such distributions.

Algorithm 1 sums up the way TDDR globally works.

5.1 Transfer Learning

Transfer Learning aims at reusing an agent competence to a task that is slightly different from the one encountered during training. A well-known example of application is to learn an agent on a simulated environment because it is impossible to do so on a "real" one (for instance in robotics) and then expose this trained agent to the real environment.

In our context, transfer learning can be a solution to overfitting: training with a real weather archive (for forecast and observation) yields inefficient neural networks when evaluated on weather data not seen during learning. We therefore propose to learn the agent on simulated weather data before evaluating it on real weather data as described in Section 4.3. For the sake of simplicity, forecast weather data $c_t^f(\bullet)$ is taken from ERA-Interim weather archives as in Section 4.3. To easily keep spatial and temporal consistency on weather data, the only simulated part is the observed weather data $c_t^o(\bullet)$ which is randomly derived from the forecast.

5.2 Weather modeling: Domain Knowledge and Domain Randomization

To make transfer learning successful, we have to design a realistic weather model. The article [6] proposes a way to simulate observed weather data $c_t^o(m)$ from the forecast $c_t^f(m)$ by adding to it a "noise" term following a zero-mean normal distribution. This normal law standard deviation varies linearly with $c_t^f(m)$: the more $c_t^f(m)$ is important, the more the standard deviation increases. However, there is little to no physical justification for such a normal law.

Furthermore, for Earth observation applications, it is not important to simulate accurately the observed cloud cover percentage $c_t^o(m)$. The only thing that matters is the probability $\mathbb{P}\left(c_t^o(m) \leq c_{max} \mid c_t^f(m)\right)$ that the mesh m will be validated if acquired during pass t. We therefore propose to base our simulated weather model on a Bernoulli distribution based on this probability. Note that this probability also depends on the time delta between the date at which the forecast is computed and the date at which the cloud cover is predicted by the forecast. Using ERA-Interim data, the most accurate weather prediction have a 6 hours delta. The assumption is made here that the satellite plan will be computed and updated in less than 6 hours before the acquisition date. To generate the observed weather, we therefore choose to assume this time delta is fixed and equal to 6 hours and we get rid of it in the notations.

The distribution we choose to use is then a Bernoulli law to determine if a mesh is validated. The only necessary information is thus the probability of this

event $f(c^f) = \mathbb{P}\left(c^o \leq c_{max} \mid c^f\right)$ which only depends on the most recent forecast cloud cover. Here are the steps used to build this distribution function.

Extract histograms from the weather archive Let $\mathcal{W}$ be the weather archive containing past observation and weather forecasts and $\mathcal{D}$ be the discrete set of dates at which this data is available. For any $d \in \mathcal{D}$ and $m \in \mathcal{M}$, a forecast $c_d^f(m)$ and the corresponding observed weather $c_d^o(m)$ is available in the weather archive. Let $b_0 = 0 < b_1 < \cdots < b_{N_b} = 1$ be a decomposition of $[0, 1]$. Because $c_d^f(m) \in [0, 1]$, we can therefore deduce the value of f on each of those subsets of $[0, 1]$, so $\forall i \in \{1, \cdots N_b\}$:

$$f\left(\frac{b_i - b_{i-1}}{2}\right) = \frac{\sum_{d\in\mathcal{D}} \sum_{m\in\mathcal{M}} \mathbb{1}\left(\left\{b_{i-1} \leq c_d^f(m) < b_i\right\} \wedge \{c_d^o(m) \leq c_{max}\}\right)}{\sum_{d\in\mathcal{D}} \sum_{m\in\mathcal{M}} \mathbb{1}\left(b_{i-1} \leq c_d^f(m) < b_i\right)}$$

Deduce a distribution function for each forecast value To define a value to $f(c^f)\ \forall c^f \in [0, 1]$, a linear interpolation is applied using the known values for each $\frac{b_i - b_{i-1}}{2}$. Regarding the value in 0 and 1, we choose to take a value of b_1 and b_{N_b-1} as close as possible to respectively 0 and 1 and take $f(0) \simeq f\left(\frac{b_1}{2}\right)$ and $f(1) \simeq f\left(\frac{1-b_{N_b-1}}{2}\right)$

Application during learning At each learning step t the neural network chooses a mesh $m \in \mathcal{M}$. The forecast used to make this choice $c_t^f(m)$ is known and $f(c_t^f(m))$ can then be computed. A uniform value between 0 and 1 is sampled. If it is inferior to $f(c_t^f(m))$, the mesh is validated otherwise it is rejected.

Create several distribution functions for randomization Instead of creating only one function from the whole weather archive, it is possible to build one for any subset of the weather archive $\mathcal{D}_l \subset \mathcal{D}$. If this subset has a sufficient amount of data, a function of probability of validation can be built $f_l(c_t^f(m))$. Let $\mathcal{L}$ be the set of subsets of weather archive used. In the experiments, for $l \in \mathcal{L}$, $\mathcal{D}_l$ can correspond to a given year in the archive, or to all the dates corresponding to a given season. This means that for $(l, l') \in \mathcal{L}^2$, $l \neq l'$, it is possible to have $\mathcal{D}_l \cap \mathcal{D}_{l'} \neq \emptyset$.

As algorithm 1 illustrates it, when the environment is reset, the function f_l used to validate meshes is changed. This is what we call domain randomization: at each episode, the environment uncertainties are generated differently.

5.3 Generalization to multiple meshes per satellite pass

To select multiple meshes per satellite pass, we propose to use a policy network that chooses only one mesh instead of a sequence of meshes but that is re-executed as many times as there are meshes to choose at each step. Between each NN execution at a given MDP step, the validation frame in the state space is updated in order to prevent the NN to pick the same mesh twice during the

Algorithm 1: TDDR training description

input : EO system description, weather archive for learning $\mathcal{W}'$ with dates $\mathcal{D}'$, and observed weather distributions $(f_l)_{l\in\mathcal{L}}$
output: Neural network N

for $it \leftarrow 1$ **to** 200×10^6 **do**
  Reset episode:
  1. Select randomly a date d $\in \mathcal{D}'$.
  2. Select a weather validation model f_l with $l \in \mathcal{L}$ (see Section 5.2)

  Execute an A2C iteration on episodes starting from d using f_l for observation generation:
  1. Collect batch of experiments (R_t, s_t, a_t, s_{t+1})
  2. Update neural network N with Adam [7] optimizer

end

same pass. This way, the NN output size is controlled and is equal to the size of $\mathcal{M}$ instead of $\mathcal{M}^{N_m}$. This has an important advantage: in a realistic case, the number of meshes per pass that can be taken for the large coverage can vary. If the actual number of meshes to take is smaller than N_m used to train the network, the one-mesh-choice neural network is more easy to adapt than the $\mathcal{M}^{N_m}$-output-size neural network: the last one would rate and return a N_m-tuple and there is no easy way to choose a subset of the N_m with the information returned by the policy and value neural networks.

6 Experiments

Based on the hypotheses from Section 4.2, we have implemented a simulator using the OpenAI Gym framework. Our goal is to compare the trained model with the reference algorithm defined in [6] on the same episodes.

6.1 Earth Observation system and scenario definition

The EO system is defined by 4 SSO (Sun Synchronous Orbit) satellites. All satellites have an altitude of 660 km and an inclination of 98.23 degree. The satellite orbits are on two different local hours to increase the chances of encountering clear sky conditions from one pass to another.

The steps of the MDP correspond to satellite passes, as explained in section 4. With the following configuration, we have an average of 4 satellites passes per day over France.

The requested area in the experiments corresponds to France. We consider satellite sensors with a swath of 60 km. We thus have $K = 122$ meshes to acquire.

We use ERA-Interim dataset [2] to obtain archived cloud cover forecasts. We use ground-truth cloud cover observation only during model validation. During training, the realistic weather distribution is used to validate the meshes selected as described in section 5.2. $N_b = 12$ bins to build the weather distribution have been used.

6.2 Reference algorithms

The benchmark of our agent is done against a simple heuristic. We use the heuristic defined in [6] with a parameter α set to 0.0. This corresponds to a heuristic that always chooses the meshes with the smallest forecasted cloud cover on the next pass. The long-term component of the heuristic did not improve results with real weather data regardless of the fine-tuning of the α and β parameters.

6.3 Train and test methodology

The A2C agent is trained using weather forecasts from 2015 and a weather observation model (see section 5.2) built using data from 2008 to 2013. We build a different weather observation model per year, for a total of 5 models that can be chosen randomly during training (see Algorithm 1). During evaluation, we use archive data from 2014 and 2016. This evaluation of separate weather year period allows to assess policy generalization.

The observation space is a concatenation of 20 frames of weather forecast for the next 20 satellites passes and the validation frame ($N_{pass} = 21$) as described in 4.2. We used the A2C implementation from the OpenAI baselines framework [3] and a convolutional network similar to the one used in [6]. We train the agent using 8 parallel environments. The training takes about 20 days using a computer with 8 vCPUs and a K80 GPU.

For evaluation, the model is transferred to the realistic environment where observations are taken directly from the ERA-Interim dataset (instead of using domain knowledge model). To compare the model with the heuristic, we run 250 episodes for each evaluation year and each acquisition capacity N_m. For a given year and acquisition capacity, those runs use a different starting date, chosen randomly, for each episode but all the methods are compared on the same dates. It aims at providing variable weather conditions throughout the year.

Furthermore, to assess the gain of domain randomization, we present results using the TD method without domain randomization. This means that there is only one distribution used to generate observed weather f, computed using all the dates $\mathcal{D}$ available in the weather archive $\mathcal{W}$.

6.4 Experimental results and discussion

The TDDR agent performs better than the heuristic in most of the cases (weather, acquisition capacity). The Figure 2 illustrates the importance of domain randomization. TD method has similar results than the heuristic one. The gain of TDDR method decreases when the satellite capacity increases. The network is trained to select a single mesh, we suspect the workaround described in Section 5.3 is not adapted to a scenario with high acquisition capacity. When satellite capacity increases (more meshes per pass), the completion duration reaches an asymptote: the bottleneck becomes the delay needed to obtain clear sky conditions in difficult areas.

The difference between Figure 2 and Figure 3 shows that the model performance is higher on cloudy weather conditions since the gain in percentage relative to the heuristic increases on years with longer episode duration. Besides,

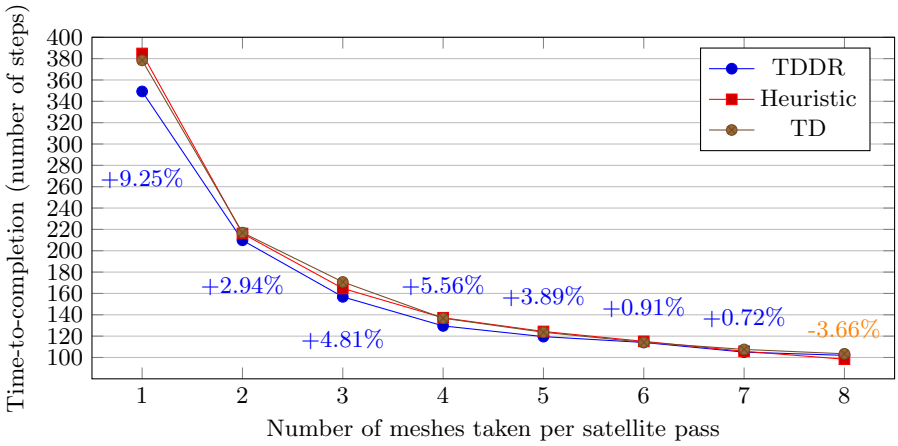

Fig. 2. France mean completion on episodes evaluated in 2014 depending on number of meshes taken per satellite pass. The percentage given is the gain of TDDR vs heuristic.

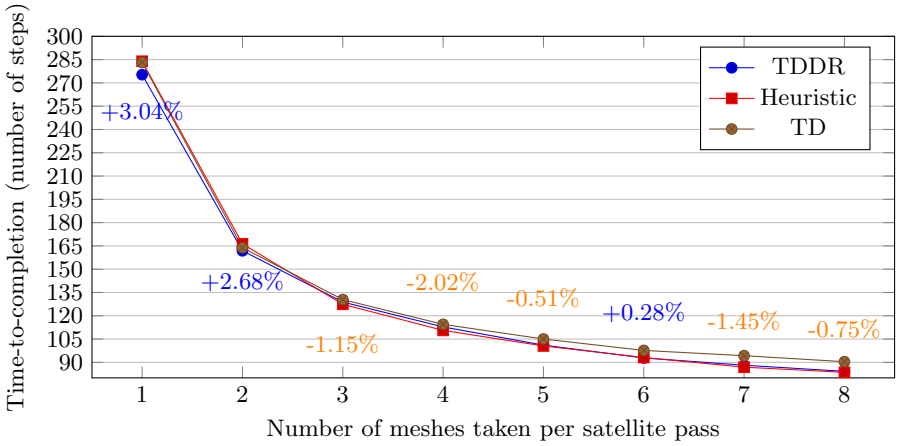

Fig. 3. France mean completion on episodes evaluated in 2016 depending on number of meshes taken per satellite pass. The percentage given is the gain of TDDR vs heuristic.

we evaluated the model on the training period (2015) and the gains observed were the same as the ones obtained on 2014 evaluation. This proves the capacity of the agent to generalize on multiple and unseen weather conditions and that there is no overfitting.

7 Conclusion

In this work, we have proposed to apply Reinforcement Learning in a realistic Earth Observation satellite scheduling environment. The TDDR method uses a mix of existing techniques to apply DRL to a real world problem. It prevents from overfitting and allows us to deal with complex action space and realistic

weather data. In a series of simulation-based experiments, our method challenges existing methods used in operational missions with a gain up to 10% compared to state-of-the-art heuristic.

Our method is however limited to a single AOI: the agent is trained using a specific configuration, and changing the AOI requires re-training the agent. In future research, we will look at combining TDDR with a multi-agent approach, where each agent would correspond to a mesh to acquire. Such combination would allow us to generalize further and train a single agent capable of handling any kind of area, with various shape and size.

References

1. Bensana, E., Verfaillie, G., Michelon-Edery, C., Bataille, N.: Dealing with uncertainty when managing an earth observation satellite. EUROPEAN SPACE AGENCY-PUBLICATIONS-ESA SP **440**, 205–210 (1999)
2. Dee, D.P., Uppala, S.M., Simmons, A.J., et al.: The ERA-Interim reanalysis: Configuration and performance of the data assimilation system. Quarterly Journal of the Royal Meteorological Society **137**(656), 553–597 (2011)
3. Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., Zhokhov, P.: Openai baselines. GitHub repository (2017)
4. Ferreira, P.V.R., Paffenroth, R., Wyglinski, A.M., Hackett, T.M., Bilén, S.G., Reinhart, R.C., Mortensen, D.J.: Multiobjective reinforcement learning for cognitive satellite communications using deep neural network ensembles. IEEE Journal on Selected Areas in Communications **36**(5), 1030–1041 (2018)
5. Gabrel, V., Murat, C.: Mathematical programming for earth observation satellite mission planning pp. 103–122 (2003)
6. Hadj-Salah, A., Verdier, R., Caron, C., Picard, M., Capelle, M.: Schedule earth observation satellites with deep reinforcement learning. IWPSS (2019)
7. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014)
8. Lemaıtre, M., Verfaillie, G., Jouhaud, F., Lachiver, J.M., Bataille, N.: Selecting and scheduling observations of agile satellites. Aerospace Science and Technology **6**(5), 367 – 381 (2002)
9. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning (2013)
10. Packer, C., Gao, K., Kos, J., Krähenbühl, P., Koltun, V., Song, D.: Assessing generalization in deep reinforcement learning (2018)
11. Povéda, G., Regnier-Coudert, O., Teichteil-Königsbuch, F., Dupont, G., Arnold, A., Guerra, J., Picard, M.: Evolutionary approaches to dynamic earth observation satellites mission planning under uncertainty p. 1302–1310 (2019)
12. Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world (2017)
13. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks (2015)
14. Wang, X., Wu, G., Xing, L., Pedrycz, W.: Agile earth observation satellite scheduling over 20 years: formulations, methods and future directions (2020)
15. Xuexuan Zhao, Zhaokui Wang, G.Z.: Two-phase neural combinatorial optimization with reinforcement learning for agile satellite scheduling. AIAA American Institute of Aeronautics and Astronautics (2020)