Deep Reinforcement Learning with Omnidirectional Images: application to UAV Navigation in Forests
Charles-Olivier Artizzu, Guillaume Allibert, Cédric Demonceaux

To cite this version:
Charles-Olivier Artizzu, Guillaume Allibert, Cédric Demonceaux. Deep Reinforcement Learning with Omnidirectional Images: application to UAV Navigation in Forests. 17th International Conference on Control, Automation, Robotics and Vision (Finalist "Best Paper Student"), Dec 2022, Singapore, Singapore. hal-03812448

HAL Id: hal-03812448
https://hal.science/hal-03812448
Submitted on 12 Oct 2022

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.
Deep Reinforcement Learning with Omnidirectional Images: application to UAV Navigation in Forests.

Charles-Olivier Artizzu*, Guillaume Allibert† and Cédric Demonceaux‡

*Université Côte d'Azur, CNRS, I3S, France. Emails: artizzu.allibert@i3s.unice.fr
† ImViA, Université Bourgogne Franche-Comté, France. mail: cedric.demonceaux@u-bourgogne.fr

Abstract—Deep Reinforcement Learning (DRL) is highly efficient for solving complex tasks such as drone obstacle avoidance using cameras. However, these methods are often limited by the camera perception capabilities. In this paper, we demonstrate that point-goal navigation performances can be improved by using cameras with a wider Field-Of-View (FOV). To this end, we present a DRL solution based on equirectangular images and demonstrates its relevance, especially compared to its perspective version. Several visual modalities are compared: ground truth depth, RGB, and depth directly estimated from these 360° RGB images using Deep Learning methods. Next, we propose a spherical adaptation to take into account the spherical distortions of omnidirectional images in the convolutional neural networks (CNNs) used in the actor-critic network and show a significant improvement in navigation performance. Finally, we modify the perspective depth estimation network using this spherical adaptation and demonstrate a further performance improvement.

Index Terms—Vision for robots, Mobile robotics, Perception systems.

I. INTRODUCTION

Depth is the most commonly used input for drone navigation in complex, unstructured environments. This visual modality enables reliable obstacle detection and thus safe trajectories for Unmanned Aerial Vehicles (UAVs). Most of today’s commercial drones use LIDARs to obtain an accurate depth estimation. However, these sensors are heavy and power-consuming. With recent accuracy improvements in monocular perspective depth estimation, RGB cameras are now being considered to replace LIDARs.

In parallel, more affordable and accurate omnidirectional cameras have recently been commercialized. The latest generation of cameras provides spherical images with up to 4K resolution and high-frequency capture. In addition, with their 360° Field-Of-View (FOV), they can capture the entire environment in a single image. Therefore, this sensor offers great potential in many robotic tasks, including drone navigation.

To solve the point-goal navigation, most state-of-the-art algorithms use a local planner based on deep reinforcement learning (DRL). This method has been proven to be effective in driving short-term navigation, and collision avoidance in many papers [1], [2], [3]. The DRL approach proposes to learn a navigation policy through trial-and-error experiments where the agent interacts with the environment based on its perception, and state [4], [5]. A reward is designed to promote or prevent specific behaviors and thus influence the agent’s policy. This method offers excellent generalization capabilities but requires a very long learning process, usually performed in virtual environments to allow thousands of trials and exploration of failure cases.

Perception is crucial for these DRL algorithms because the agent selects its next action based only on what it observes. Therefore, this information must be as accurate as possible. In recent years, monocular depth estimation in perspective images has seen significant progress, reaching a very high accuracy [6], [7], [8].

However, when applied directly to omnidirectional images, these perspective methods suffer from a domain shift. Indeed, these images present large distortions near the polar regions, which significantly impact the accuracy of networks designed for perspective images. Some specialized networks have been proposed to estimate depth in spherical images [9], [10]. In this paper, we offer an alternative method based on convolutions taking into account distortions [11].

The contributions of this paper are fourthfold:

- First, we propose a point-goal navigation DRL solution based on 360° FOV images as input.
- Second, we compare the performance of this solution with its baselines using perspective images. To our knowledge, this is the first comparison of omnidirectional and conventional images for DRL. The obtained results confirm the advantage of omnidirectional systems.
- Third, the ground truth depth, RGB, and estimated depth modalities are compared in a performance benchmark. This comparison reveals that using the MIDAS network [8] is more reliable than methods based on RGB alone.
- Fourth, we propose an adaptation to take into account the spherical distortions of equirectangular images. We implement this modification in two different locations. One in the Actor-Critic network proves that spherical networks’ performance is improved when trained with a specific spherical adaptation. The other directly in the MIDAS network demonstrates that a pre-trained network can be adapted to equirectangular images without additional training.

The structure of this paper is the following. Section II describes the goal-driven navigation solution adopted here. Next, Section III presents the FOV and modality performance benchmark. Finally, Section IV presents the proposed adaptation for spherical images and the associated results.
II. DRL-BASED NAVIGATION SOLUTION

Point-goal navigation can be modeled by a Markov decision process (MDP). In this formulation, the agent interacts with the environment by performing actions following a specific policy in a given state. The agent receives a positive or negative reward, to promote or prevent certain behaviors. In our specific case, the drone is rewarded positively when it reaches its goal and negatively when it collides with obstacles, gets stuck in a loop, or moves too far away from its objective.

A. State

In DRL, the agent uses its current state to determine its following action. Therefore, the observation of his state must be accurate and complete enough to provide sufficient information to make an appropriate decision. But in return, a too exhaustive state will overload the agent with redundant parameters. Thus, in this study, we propose to use a state containing bare minimum information to achieve the two main objectives: point-goal navigation and obstacle avoidance.

For the navigation task, only the relative distance and direction of the current goal are provided. At each time-step \( t_k = k \Delta t \), where \( \Delta t \) is the control sampling time, we define the distance \( d_k \) and the angle \( \theta_k \) to the goal:

\[
\begin{align*}
    d_k &= \| P_k - P^* \|_2, \\
    \theta_k &= \arctan2(y^* - y_k, x^* - x_k) - \psi_k,
\end{align*}
\]

(1)

with a drone at position \( P_k = (x_k, y_k, z_k) \) heading towards a fixed goal at position \( P^* = (x^*, y^*, z^*) \) with a yaw angle \( \psi_k \). At the same time, the drone captures its surroundings with its perception sensor and transforms it into an image noted \( I_k \). Finally the state of the drone is defined at each time-step \( t_k \) by: \( S_k = [d_k, \theta_k, I_k] \).

B. Visual modalities compared

We propose to compare three different visual modalities for the \( I_k \) image: Ground Truth (GT) depth, RGB, and Estimated Depth (ED) using Deep Learning. In parallel, two different capture FOVs are compared for all these modalities: a limited one (90°) and an omnidirectional one (360°). To compare these features fairly, we keep the same network architecture between the different study cases. Thus, the image resolution \( I_k \) is independent of the modality or FOV and fixed at 100 × 100 pixels. Fig. 1 shows the different inputs used in this paper.

C. Action

In forest point-goal navigation, the drone must navigate between trees without collision. Due to the mostly vertical structure of the trees, we propose to realize obstacle avoidance in an iso-altitude plane. Therefore, the drone controller keeps the drone’s altitude constant during its flight while the pilot focuses on rotational movements. The agent moves by selecting an action \( a_k \in [-\pi, \pi] \) that corresponds to a desired rotation angle among \( N_b \) directions, which is later sent to the low-level controller [12]:

\[
a_k = -\pi + \frac{2i}{N_b - 1} \pi \quad i \in \{0, ..., N_b - 1\}. 
\]

(2)

D. Network and Reward

In this paper, we focus on the perception of the DRL algorithm using different FOVs and modalities as input. Therefore, we have based ourselves on the rest of the DRL solution from already proven contributions, particularly for the actor-critic network and reward function.

The PPO algorithm [13] processes the agent state using an actor-critic network to predict the next best action. The architecture proposed in this paper is based on contributions that have already proven effective for UAV navigation [14], [2], [4]. First, the visual part \( I \) of the state \( S \) is preprocessed using a succession of convolutions (CONV) and fully connected networks (FC). The resulting 32-dimensional vector is then combined with the goal information \((d_k, \theta_k)\) to determine the next action \( a_k \) using another fully connected network (RFC). The global network is shown in Fig. 2 and requires 60k parameters (more details in Appendix A).

The reward function is crucial in DRL. It describes how the agent performed during the episode. In [15], the authors propose a solution that shows promising results in goal-driven navigation and obstacle avoidance. Their reward function is ideally suited for our problem. However, they use depth sensors in their model to maintain a safe distance to the obstacles during the flight. In order to be more independent to the modalities where absolute depth is not necessarily known, we have removed this safe distance in the reward function.

At each time-step \( t_k \), the reward is computed using the relative goal state \((d_k, \theta_k)\), a penalization term \((-0.02)\), and an...
additional termination reward if the agent reached an terminal state $R_{end}$. The resulting function is:

$$R_k = -0.1d_k - 0.05\|\theta_k\| - 0.02 + R_{end}.$$  
(3)

Three different terminal states are possible for each episode:

- the drone collides with an obstacle, so it receives a large negative reward of $R_{end} = -5$;
- the drone reaches its goal ($d_k < 1m$), so it receives a great positive reward of $R_{end} = 5$;
- the drone is stuck in a loop (more than 200 time-steps) or going too far away from its goal ($d_k > 100m$), so it receives a small negative reward of $R_{end} = -2$.

III. FIRST RESULTS

We train and compare the performances of our proposed DRL solution in a simplified forest environment RDMAP. We choose Unreal Engine [16] as rendering software to build this environment. Connected to AirSim [12], an open-source robotics plugin, the simulator can provide high-fidelity image capture and a low-level controller to stabilize a drone.

The training is performed on Nvidia Tesla-V100 graphic cards and takes about 8 to 10 hours. Inference on 300 drone trajectories takes between 120 and 180 minutes.

A. Visual modalities compared

As presented in Section II-B, three different modalities are compared in this article: ground truth depth, RGB and estimated depth with Deep Learning. In parallel, for each modality, two FOVs are studied: one perspective and one omnidirectional.

The ground truth depth and RGB images are directly provided by the AirSim simulator. The GT depth value is cropped to the [0..5] meters range to remain representative of the capabilities of a small UAV-mounted LIDAR.

To estimate depth from equirectangular RGB images, we can either use specialized spherical networks or standard perspective solutions with spherical adaptations. Several omnidirectional networks have been published, such as [9], [10]. However, they require significant computational power, and their specific training on omnidirectional datasets makes them less oriented towards drone navigation. Therefore, we choose a lightweight perspective network MIDAS [8]. This Convolutional Neural Network (CNN) is one of the lightest and most accurate published perspective depth estimation networks, with already proven performance in several mobile and UAV applications [17], [18]. We directly use its pre-trained version midas_v21_small to be representative of an embedded drone solution. Solutions to adapt to equirectangular images are proposed in Section IV.

B. Simplified training and testing environment RDMAP

We build a simplified forest environment named RDMAP. This $200 \times 200$ meters terrain is made of many vertical cylinders placed randomly and schematizing a dense forest of tree trunks. Fig. 3 shows an overview of this simplified environment.

C. Training

The proposed DRL solution is trained in RDMAP using the different visual modalities and FOVs. Each training takes 100k time-steps and uses the same schedule and parameters for a fair comparison. Appendix A shows the hyperparameters used to tune the PPO algorithm. During training, the distance between the initial position of the drone and its target is always 20 meters.

D. Metrics

Two standard metrics are used to evaluate each solution navigation results:

- the Success Rate $SR$ (the percentage of successful runs divided by the total number of runs) to directly assess the drone’s abilities to reach its goal ($d_k < 1m$);
- the Success weighted by Path Length, as defined in [19]:

$$SPL = \frac{1}{N} \sum_{i=1}^{N} S_i \frac{\ell_i}{p_i},$$

where $S_i$ is 0 or 1 depending on the success of the episode, $p_i$ is the length of the drone’s trajectory and $\ell_i$ is the shortest distance between the initial and goal points. Thus, the closer the drone trajectory is to the shortest path, the closer the SPL is to 100%. Besides, failed tests are strongly penalized by the boolean value $S_i$.

E. Inference in RDMAP

We test the performance of the trained DRL solution directly in the training environment RDMAP. The testing process is identical for each modality or FOV; point-goal navigation is proven on the same 600 randomly selected drone-goal pairs on the map. However, unlike training, the distances between the targets are not only 20m: we also used distances of 40m and 60m to test our solution and prove its robustness thoroughly.

As presented before in Section II-B, three visual modalities (GT depth, RGB, and Estimated Depth (ED) with MIDAS) and two FOVs (perspective and omnidirectional) are compared. The results are presented below in Table I.

Solutions using omnidirectional images consistently outperform those using perspective ones. The larger FOV improves navigation and obstacle detection tasks. Obstacles invisible in a narrow FOV are now detected in the omnidirectional images reducing the number of collisions (higher SR). Moreover, the
these reasons, we choose to implement this distortion-aware convolution using a pre-computed offsets table.

During training and inference of the DRL solution, these offsets are directly applied to change the point position in every kernel to project the spherical image locally onto its perspective equivalent. For example, Fig. 4 shows different distortion-aware kernel shapes in function of their position in the equirectangular image.

B. Actor-Critic Network Adaptation

The four convolutional layers of the proposed actor-critic network architecture are modified. Prior to training, the offsets tables are computed based on the resolution of the observation used and the different parameters of each adapted convolutional layer. Fig. 5 shows how this additional plugin is applied on the network architecture.

IV. DRL SPHERICAL ADAPTATION

The proposed DRL-based navigation solution shows promising results using 360° FOV observations as inputs. However, all spherical projections introduce some distortions. In particular, equirectangular images show significant distortions near the polar regions.

Therefore, we present a solution to consider these distortions by maintaining the local coherence of pixels during convolution operations. This approach is applied in two different locations in our proposed solution. First, the convolution layers of the actor-critic network used in the DRL algorithm are modified. Second, the MIDAS network is adapted to improve depth prediction in equirectangular images without additional training.

![Fig. 4. Example of kernels with different latitude and longitude. In blue is the center of the kernel, in green the perspective kernel and in red the adapted equirectangular one. The wider distortions are near the poles.](image)

### TABLE I

<table>
<thead>
<tr>
<th>RUN</th>
<th>SR (%)</th>
<th>SPL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>90° FOV GT depth</td>
<td>76.0</td>
<td>54.0</td>
</tr>
<tr>
<td>360° FOV GT depth</td>
<td>88.8</td>
<td>68.6</td>
</tr>
<tr>
<td>90° FOV RGB</td>
<td>68.3</td>
<td>52.8</td>
</tr>
<tr>
<td>360° FOV RGB</td>
<td>85.7</td>
<td>62.6</td>
</tr>
<tr>
<td>90° FOV ED</td>
<td>69.7</td>
<td>57.9</td>
</tr>
<tr>
<td>360° FOV ED</td>
<td>86.0</td>
<td>67.0</td>
</tr>
</tbody>
</table>

### TABLE II

<table>
<thead>
<tr>
<th>RUN</th>
<th>SR (%)</th>
<th>SPL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>360° FOV GT depth (baseline)</td>
<td>88.8</td>
<td>68.6</td>
</tr>
<tr>
<td>360° FOV GT depth (DRL adapted)</td>
<td>94.8</td>
<td>78.3</td>
</tr>
<tr>
<td>360° FOV ED (baseline)</td>
<td>86.0</td>
<td>69.0</td>
</tr>
<tr>
<td>360° FOV ED (DRL adapted)</td>
<td>89.5</td>
<td>73.3</td>
</tr>
</tbody>
</table>

Results in RDMAP: The equirectangular adapted DRL solution is tested in the RDMAP environment in a process similar to that presented in Section III-C.

### Results in RDMAP

For each modality, the distortion-aware (adapted) solutions show highly better performance. Maintaining local pixel coherence during convolutions helps the Actor-Critic network to better detect obstacles in the $I_k$ input image. As a result, the drone trajectories are better optimized, with fewer collisions and faster paths. In particular, it allows the spherically adapted ED solution to outperform the non-adapted GT depth based solution, especially in trajectory optimization.

C. MIDAS network Adaptation

To further improve the results of the previous section, we propose a fast and elegant method to increase depth estimation accuracy in equirectangular images without additional training.

![Fig. 5. Pre-computed spherical adapted offsets are added to the four convolution layers of the Actor-Critic network during training and testing of the DRL-solution.](image)
MIDAS is a lightweight depth estimation network trained on perspective datasets. Therefore, its accuracy suffers from spherical distortion when applied directly to equirectangular images. The previous test section showed that the DRL solution based on 360° estimated depth performs well but is still lower than ground truth depth performances.

Therefore, we propose to improve depth prediction accuracy by applying a spherical adaptation to the MIDAS network. The decoder convolutions are modified with additional pre-computed offsets as explained in Section IV-A. No additional training is performed on an equirectangular dataset as we keep the training weights of the MIDAS authors on perspective datasets.

The actor-critic network spherical adaptation remains active as it has shown excellent results in the previous section. Table III shows the performances.

### Results in RDMAP

The DRL-solution using MIDAS network adapted for equirectangular images is trained and tested in the RDMAP environment. Table III shows the performances.

<table>
<thead>
<tr>
<th>RUN</th>
<th>SR (%)</th>
<th>SPL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>360° FOV ED (DRL &amp; MIDAS adapted)</td>
<td>89.8</td>
<td>74.5</td>
</tr>
</tbody>
</table>

**TABLE III**

PERFORMANCES IN RDMAP.

The performance of our solution with the spherical DRL and MIDAS adaptation is better than that of the DRL-only adapted solution. As a result, the MIDAS adaptation has better depth estimation accuracy than its baseline on equirectangular images.

The depth is improved locally with better obstacle detection and thus a higher success rate. In parallel, the estimation is also globally enhanced, allowing a better understanding of the complete environment of the UAV and faster trajectories (higher SPL).

Therefore, using a larger FOV and spherical adaptations allows the final version of our solution based on estimated depth to go from an initial success rate of 70% to a final version close to 90%. Moreover, the drone trajectories also gain noticeably in speed.

### V. GENERALIZATION TO A PHOTOREALISTIC FOREST ENVIRONMENT

To reduce the gap with reality, we build the RWFOREST forest environment. Using the best rendering capabilities of Unreal Engine and forest textures from its marketplace, we create a photorealistic forest with complex lighting. Unlike the simplified trunks in RDMAP, the trees here have different sizes, branches, and dense foliage. As a result, RWFOREST is a challenging environment where the captured images $I_k$ are much more complex to analyze and translate into actions. Fig. 7 shows some observations from this environment.

![RWFOREST](image)

**Fig. 7.** RWFOREST: (Left: RGB, Middle: depth, Right: MIDAS depth [8]).

We directly test the previously trained solutions in this new environment to challenge the robustness of our proposed solution. No additional training was performed in RWFOREST. Instead, the pre-trained models of the section III-C were directly tested on 600 objectives of 20m, 40m, and 60m. The Table IV presents the result of the best performing solutions from the previous sections: the spherical adapted ground truth and the DRL & MIDAS adapted estimated depth.

<table>
<thead>
<tr>
<th>RUN</th>
<th>SR (%)</th>
<th>SPL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>90° FOV GT depth</td>
<td>89.7</td>
<td>76.9</td>
</tr>
<tr>
<td>360° FOV ED (DRL &amp; MIDAS adapted)</td>
<td>84.7</td>
<td>62.1</td>
</tr>
</tbody>
</table>

**TABLE IV**

PERFORMANCES IN RWFOREST.

Although a challenging test using much more complex images than those used for training, our proposed solution still performs very well. The DRL & MIDAS adapted 360° FOV estimated depth reaches almost 85% success. This proves the robustness of our proposed solution to domain change despite the increase in observational complexity.

The MIDAS network also proves its stability in predicting consistent depth estimation in various environments without additional training. Despite its small size, it is reliable enough to build a DRL navigation solution with performance close to ground truth modalities.

### VI. CONCLUSION

This paper presented a point-goal navigation solution for drones in forests using monocular omnidirectional images. We have shown that this equirectangular solution outperforms its perspective reference. Indeed, thanks to a larger FOV, the drone has a better understanding of its environment, resulting in fewer collisions and better trajectory optimization.
In parallel, we have studied a DRL solution using monocural estimated depth as visual input and compared it to ground truth depth and RGB. Despite significant spherical distortions, the MIDAS network trained on perspective images provides reliable and accurate depth prediction, sufficient to outperform a DRL solution using only RGB.

Furthermore, we have presented a fast and elegant spherical adaptation that aims to maintain local pixel coherence during convolution operations without additional training. Again, the point-goal navigation task has been further improved by implementing this adaptation in the Actor-Critic network and MIDAS. Finally, even when tested in a new and more complex environment, our solutions showed high robustness to domain shift.

Our code implementation will be open-source and available on GitHub at https://github.com/COATZ/OMNI-DRL.

ACKNOWLEDGMENT

This work was supported by the ANR CLARA project, grant ANR-18-CE33-0004 of the French Agence Nationale de la Recherche. This work was granted access to the HPC resources of IDRIS under the allocation AD011013128 made by GENCI.

REFERENCES


APPENDIX

<table>
<thead>
<tr>
<th>LAYER</th>
<th>TYPE</th>
<th>NB_PARAMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC1</td>
<td>FC(8<em>7</em>7, 64)</td>
<td>25152</td>
</tr>
<tr>
<td>FC2</td>
<td>FC(64, 32)</td>
<td>2080</td>
</tr>
<tr>
<td>RF1C</td>
<td>FC(34,64)</td>
<td>2172</td>
</tr>
<tr>
<td>RF2C</td>
<td>FC(64,128)</td>
<td>8320</td>
</tr>
<tr>
<td>RF3C</td>
<td>FC(128,128)</td>
<td>16512</td>
</tr>
<tr>
<td>RF4C</td>
<td>FC(128,37)</td>
<td>4773</td>
</tr>
</tbody>
</table>

TABLE V CUSTOMCNN

Total: 60814 parameters.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>learning rate</td>
<td>0.0005</td>
</tr>
<tr>
<td>number of steps</td>
<td>2048</td>
</tr>
<tr>
<td>batch size</td>
<td>64</td>
</tr>
<tr>
<td>number of epochs</td>
<td>10</td>
</tr>
<tr>
<td>gamma</td>
<td>0.99</td>
</tr>
<tr>
<td>Δt</td>
<td>200 ms</td>
</tr>
<tr>
<td>N_a</td>
<td>57 actions</td>
</tr>
</tbody>
</table>

TABLE VI SIMULATION HYPERPARAMETERS