AI-driven Design of Novel Vehicles Using Reinforcement Learning

Krishiv Agarwal, Adam Cobb, Daniel Elenius, Anirban Roy, and Susmit Jha
Computer Science Laboratory
SRI International
{adam.cobb, daniel.elenius, anirban.roy, susmit.jha}@sri.com

Abstract

Computer-aided design (CAD) is a promising new area for the application of artificial intelligence (AI) and machine learning (ML). The current practice of design of cyber-physical systems uses the digital twin methodology, wherein the actual physical design is preceded by building detailed models that can be evaluated by physics simulation models. These physics models are often slow and the manual design process often relies on exploring near-by variations of existing designs, stifling creativity. Coming up with novel designs is an art requiring very high level of expertise. AI holds the promise of breaking design silos and increasing the diversity and performance of designs by accelerating the exploration of the design space. In this paper, we focus on the design of land vehicles. We develop a designer using reinforcement learning that synthesizes novel vehicle designs and present results showing how AI can produce novel and creative designs. We will demo our work in the workshop.

1 Creative Designs Using Artificial Intelligence

Artificial intelligence (AI) is transforming multiple application domains, especially those pertaining to areas of creativity and design. Examples of this success can be seen in both art and music Briot et al. [2020], Cetinic and She [2022], Ramesh et al. [2021], Razavi et al. [2019], with particularly impressive results in text-to-image generation Ramesh et al. [2022], Saharia et al. [2022], showing the potential of human-AI interaction for creative design. However, in comparison to domains such as text-to-image generation, in scientific design contexts (such as unmanned air vehicles and cyber-physical systems, in general) the requirements of the final design are much more structured and subject to the satisfaction of correctness requirements and optimization of performance objectives. These domains include program synthesis, drug discovery, architecture, and the design of mechanical components Parisotto et al. [2016], Korshunova et al. [2021], Buonamici et al. [2020], Kołata and Zierke [2021], Granados-Ortiz and Ortega-Casanova [2021]. Unlike the generation of images and text, in these structured-sensitive domains, even generating valid artifacts is nontrivial. In recent work Cobb et al. [2023] [1], we have demonstrated how we can use AI to create novel drone designs. In this paper, we focus on the design of land vehicles using reinforcement learning.

Our focus is on discovering novel vehicle designs that are more suitable for a new environment and exploit novel characteristics such as non-cylindrical wheels and non-rigid chassis. We use Webot - a high-fidelity physics model for simulating the vehicle design and ensuring the created design is not only novel in appearance but exhibits the expected behavior. The use of machine learning in computer aided design (CAD) has gained significant attention but existing work is limited to low-complexity designs such as SketchGraphs dataset Seff et al. [2020], simple robotic arms Xu et al. [2021] and airfoil design Chen et al. [2022]. We emphasize that we aim at novelty of design not just in their appearance but also in their functionality.

[1]https://github.com/SRI-CSL/AircraftVerse

2 Vehicle Design Space and RL-based Exploration

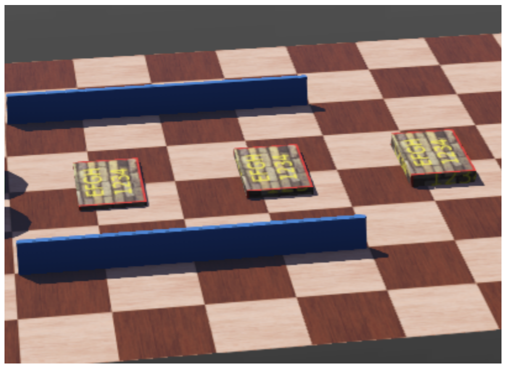

Figure 1: Example of obstacle course: Our goal is to design novel vehicles that can complete this course successfully and quickly.

For our vehicle design problem, we consider an obstacle course consisting of a race track with multiple obstacles in it of varying height and size as shown in Figure 1. This environment and the vehicle under design are instantiated in a high-fidelity physics engine Webot [2]. The goal is to design a vehicle that can finish this track successfully and quickly. So, we set the reward function to be the sum of rewards obtained by crossing each of the obstacles discounted by the total time taken to complete the race track. We use a probabilistic grammar to specify the space of possible designs. This grammar provides several avenues for novelty in the design of vehicle, and these include: (a) allowing chassis to be non-rigid and comprising several components connected via flexible joints (b) wheels that can be of different sizes and shapes such as cylindrical, spherical or square, and (c) varying offsets of wheels. We present a brief definition of the design space in Appendix A of Supplementary Material. For exploring the design space and constructing novel designs, we use reinforcement learning (PPO Schulman et al. [2017]). This is a preliminary investigation into design of novel land vehicles inspired by our recent work on novel drone design Cobb et al. [2023] (Appendix B of Supplementary Material). This can be useful in niche applications such as space exploration which still relies on manual ingenuity for vehicle design (e.g. EELS [3]).

3 Results

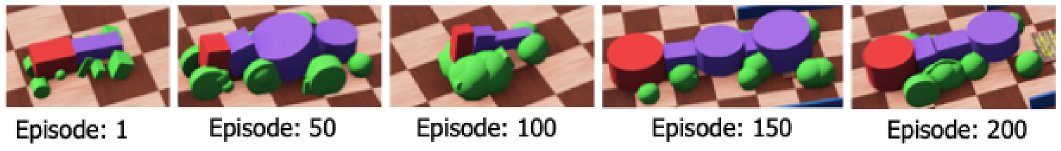

Figure 2: Examples of Vehicle Designs in Webot generated by RL in different episodes

We observe that the RL exploration stops using square wheels or cylindrical wheels, and starts mostly using sphere wheels. Further, it prefers using large cylinder as the base chassis design and adds a number of chassis segments to improve the vehicle's ability to climb over obstacles. The vehicles start resembling worms or eels as the search progresses. We illustrate some of the examples of the vehicles in different episodes in Figure 2. We also show how the mean and standard deviation of the reward evolves over the episodes. The reward stabilizes around 400 episodes.

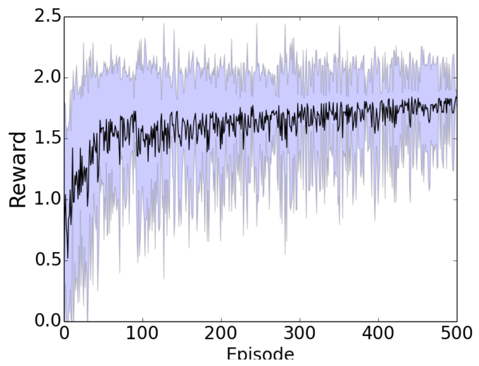

Figure 3: Improvement in reward with episodes: mean reward and the standard deviation

4 Conclusion

We have developed an AI approach that combines probabilistic programming and reinforcement learning to generate novel designs of vehicles using Webot physics simulation engine. In recent work, we also produced novel drone designs Cobb et al. [2023]. We will demonstrate our AI-driven creation of novel designs and video of the behavior of these designs in the workshop.

[2]https://cyberbotics.com/

[3]https://www.jpl.nasa.gov/robotics-at-jpl/eels

References

Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. *Deep learning techniques for music generation*, volume 1. Springer, 2020.

Francesco Buonamici, Monica Carfagni, Rocco Furferi, Yary Volpe, and Lapo Governi. Generative design: an explorative study. *Computer-Aided Design and Applications*, 18(1):144–155, 2020.

Eva Cetinic and James She. Understanding and creating art with ai: Review and outlook. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 18(2):1–22, 2022.

Qiuyi Chen, Phillip Pope, and Mark Fuge. Learning Airfoil Manifolds with Optimal Transport. In *AIAA SCITECH 2022 Forum*, page 2352, 2022.

Adam D. Cobb, Anirban Roy, Daniel Elenius, F. Michael Heim, Brian Swenson, Sydney Whittington, James D. Walker, Theodore Bapty, Joseph Hite, Karthik Ramani, Christopher McComb, and Susmit Jha. AircraftVerse: A Large-Scale Multimodal Dataset of Unmanned Aerial Vehicle Designs. *37th Conference on Neural Information Processing Systems (NeurIPS 2023)*, 2023.

F-J Granados-Ortiz and J Ortega-Casanova. Machine learning-aided design optimization of a mechanical micromixer. *Physics of Fluids*, 33(6):063604, 2021.

Joanna Kołata and Piotr Zierke. The decline of architects: Can a computer design fine architecture without human input? *Buildings*, 11(8):338, 2021.

Maria Korshunova, Boris Ginsburg, Alexander Tropsha, and Olexandr Isayev. Openchem: A deep learning toolkit for computational chemistry and drug design. *Journal of Chemical Information and Modeling*, 61(1):7–13, 2021.

Emilio Parisotto, Abdel-rahman Mohamed, Rishabh Singh, Lihong Li, Dengyong Zhou, and Pushmeet Kohli. Neuro-symbolic program synthesis. *arXiv preprint arXiv:1611.01855*, 2016.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. *arXiv preprint arXiv:2205.11487*, 2022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Ari Seff, Yaniv Ovadia, Wenda Zhou, and Ryan P Adams. Sketchgraphs: A large-scale dataset for modeling relational geometry in computer-aided design. *arXiv preprint arXiv:2007.08506*, 2020.

Jie Xu, Tao Chen, Lara Zlokapa, Michael Foshey, Wojciech Matusik, Shinjiro Sueda, and Pulkit Agrawal. An End-to-End Differentiable Framework for Contact-Aware Robot Design. In *Robotics: Science and Systems*, 2021.

Supplementary Material: AI-driven Design of Novel Vehicles Using Reinforcement Learning

A Design Space of the Vehicle

```python
@dataclass
class Wheel(ABC):
    x_offset: float = field(metadata={'range': (-1.,1.)})     # relative to vehicle length, from the center
    y_offset: float = field(metadata={'range': (0.,.05)})     # wheels can stick out from the sides a bit
    z_offset: float = field(metadata={'range': (-.01,.05)})     # wheels can stick out from the bottom a bit

    def to_lowlevel(self, index: int, body_size: Tuple[float,float,float]):
        x_off = (body_size[0] / 2.) * self.x_offset
        y_off = (body_size[1] / 2.) + self.y_offset
        z_off = - ((body_size[2] / 2.) + self.z_offset)
        return { "Name": f"WHEEL{index+1}", "WheelNum": f"{index+1}", "Symmetrical": True, "Anchor": f"{x_off} {y_off}
        ↪ {z_off}" }

@dataclass
class SquareWheel(Wheel):
    size1: float = field(metadata={'range': (0.01,0.1)})
    size2: float = field(metadata={'range': (0.01,0.1)})
    size3: float = field(metadata={'range': (0.01,0.1)})

    def to_lowlevel(self, idx, body_size: Tuple[float,float,float]):
        d = super().to_lowlevel(idx, body_size)
        return { **d, "Shape": "Box", "Dimensions": { "size": f"{self.size1} {self.size2} {self.size3}" } }

@dataclass
class CylinderWheel(Wheel):
    height: float = field(metadata={'range': (0.01,0.1)})
    radius: float = field(metadata={'range': (0.01,0.1)})

    def to_lowlevel(self, idx, body_size: Tuple[float,float,float]):
        d = super().to_lowlevel(idx, body_size)
        return { **d, "Shape": "Cylinder", "Dimensions": { "height": str(self.height), "radius": str(self.radius) } }

@dataclass
class SphereWheel(Wheel):
    radius: float = field(metadata={'range': (0.01,0.1)})

    def to_lowlevel(self, idx, body_size: Tuple[float,float,float]):
        d = super().to_lowlevel(idx, body_size)
        return { **d, "Shape": "Sphere", "Dimensions": { "radius": str(self.radius), "subdivision": "1" } }

@dataclass
class Body(ABC):
    wheels: List[Wheel] = field(metadata={'length': (1,4)})
    rigid: bool

    @abstractmethod
    def length(self):
        pass

    @abstractmethod
    def width(self):
        pass

    @abstractmethod
    def height(self):
        pass

    def to_lowlevel(self, index: int, anchor_offset_x: float, wheel_num_offset: int):
        if index == 0:
            name = "BODY"
        else:
            name = f"BODY{index}"
        return { "Name": name, "Rigid": self.rigid, "Anchor": f"{anchor_offset_x} 0.0 0.0",
                "wheels": [w.to_lowlevel(i+wheel_num_offset, (self.length(), self.width(), self.height())) for i,w in
               ↪  enumerate(self.wheels)] }

@dataclass
class BoxBody(Body):
    size_x: float = field(metadata={'range': (0.05,0.2)})
    size_y: float = field(metadata={'range': (0.05,0.2)})
    size_z: float = field(metadata={'range': (0.05,0.2)})

    def length(self):
        return self.size_x

    def width(self):
        return self.size_y

    def height(self):
        return self.size_z

    def to_lowlevel(self,idx,offset,wheel_num_offset):
        d = super().to_lowlevel(idx,offset,wheel_num_offset)
        return {**d, "Shape": "Box", "Dimensions": { "size": f"{self.size_x} {self.size_y} {self.size_z}" } }

@dataclass
class CylinderBody(Body):
    size_z: float = field(metadata={'range': (0.05,0.2)})
    radius: float = field(metadata={'range': (0.05,0.2)})

    def length(self):
        return self.radius * 2

    def width(self):
        return self.radius * 2

    def height(self):
        return self.size_z

    def to_lowlevel(self,idx,offset,wheel_num_offset):
        d = super().to_lowlevel(idx,offset,wheel_num_offset)
        return {**d, "Shape": "Cylinder", "Dimensions": { "height": self.size_z, "radius": self.radius } }

@dataclass
class Car:
    bodies: List[Body] = field(metadata={'length': (1,4)})

    def to_lowlevel(self):
        body_offsets = [0.0]
        wheel_num_offsets = [0]
        #if len(self.bodies) > 1:
        #    body_offsets.append((self.bodies[0].length() + self.bodies[1].length()) / 2.)
        #    wheel_num_offsets.append(len(self.bodies[0].wheels))
        for i in range(1,len(self.bodies)):
            body_offsets.append(body_offsets[i-1] + (self.bodies[i-1].length() + self.bodies[i].length()) / 2.)
            wheel_num_offsets.append(wheel_num_offsets[i-1] + len(self.bodies[i-1].wheels))
        return { "bodies": [b.to_lowlevel(i,o,w) for i,(b,o,w) in enumerate(zip(self.bodies, body_offsets,
        ↪   wheel_num_offsets))] }
```

B Novel Drone Designs

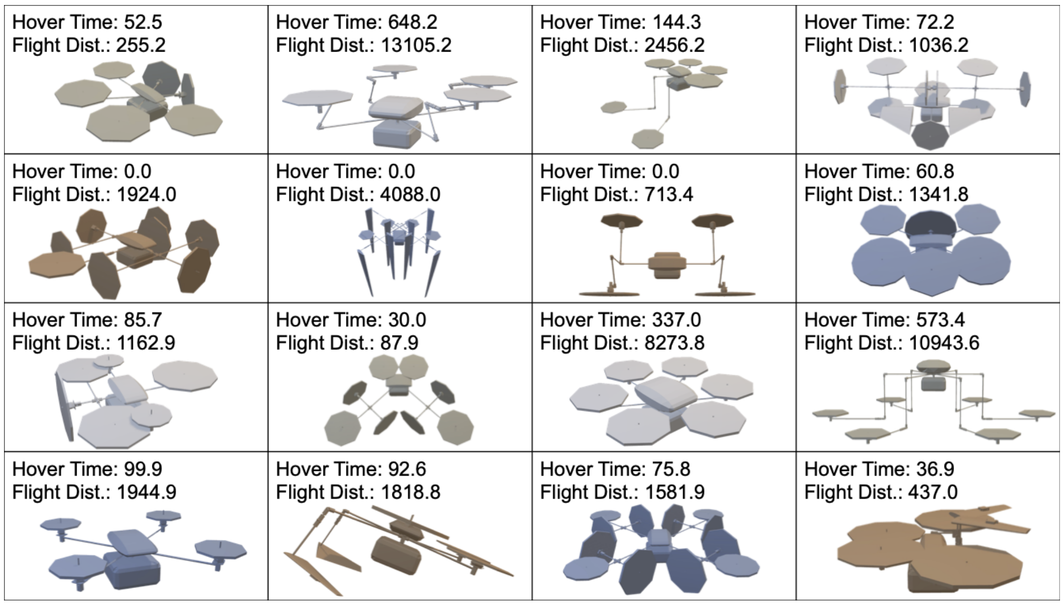

Figure 4: Examples of Drone Designs that we generated in our recent work Cobb et al. [2023]. The designs are not just novel in appearance but also exhibit different behaviors. In the figure below, we have annotated the designs with their total flight distance and hover time computed using high-fidelity physics engines. We have created a dataset of over 27000 such novel drone designs with associated CAD files such as STL and STeP, and more detailed design description including propulsion subsystem details such as battery, motor and propeller choices. This dataset is publicly available at `http://doi.org/10.5281/zenodo.6525446` with associated code at `https://github.com/SRI-CSL/AircraftVerse`.