Comparing Strategies for Visualizing the High-Dimensional Exploration Behavior of CPS Design Agents

Akash Agrawal
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA USA
aragrawa@andrew.cmu.edu

Christopher McComb
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, PA USA
ccm@cmu.edu

***Abstract*—The design of cyber-physical systems often involves search within high-dimensional design spaces. When evaluating the performance of algorithms in tasks such as these, the patterns of exploration are often informative and can help support algorithm selection. However, accurately representing these patterns in a way that is human understandable while still preserving the nuanced search complexities in the high-dimensional space is non-trivial. This work specifically examines approaches for visualizing the search trajectories of reinforcement learning agents. We assess trajectories on two exemplar problems: the design of a racecar and the design of an aerial vehicle. We compare and contrast the visualizations produced using PCA, t-SNE, UMAP, TriMap, and PaCMAP. Future work should extend this comparison to a wider variety of exemplar design problems and consider the additional challenges posed by set-based design algorithms (e.g., genetic algorithms, particle swarm optimization).**

***Index Terms*—reinforcement learning, visualization, dimensionality reduction, design, cyber-physical systems**

I. Introduction

Cyber-Physical System (CPS) design involves high-dimensional design representations and search spaces [1]–[3], making it challenging to interpret the behavior of the underlying computational design algorithms. However, the visualization of the search behavior can be helpful to understand the characteristics of these algorithms. For instance, different trajectories passing through several moderate quality regions in the early stages of exploration reflects higher breadth of search; longer trajectories traversing through very high-quality regions reflects a high depth of search; different trajectories converging to different high quality regions reflects a higher diversity in the exploration. Further, the knowledge of exploration behavior gained from a visualization can potentially be used to improve an algorithm. For instance, if a broad search pattern is leading to sub-optimal designs, one could increase the search trajectory length or tune the algorithm parameters to encourage more exploitation and less exploration. Moreover, visualization of trajectories becomes paramount in importance in a human-AI hybrid design approach, where human and AI agents collaborate in the search for solutions. In that context, communication between human and AI is known to be critical [4], [5]. Search trajectories carry information that may help a human partner better understand the types of designs that a collaborating AI agent is exploring.

The challenges of making machine learning algorithms more human-interpretable have long been recognized by the community [6], [7]. Recent work has specifically started to address the challenge posed in this work - that of visualizing the trajectories of Reinforcement Learning (RL) agents as they explore a space for solutions. Some approaches plot the agent experience using visualization methods that cluster similar state-action-reward-state quartets [8]. Recent work has also performed user studies, showing that even rudimentary visualizations can help RL practitioners to understand, diagnose, and begin to improve the implementation of RL algorithms [9]. Other work has appealed to proven data visualization methods in order to help make RL information more interpretable [10]. However, much of this prior work addressed relatively low-dimensional RL domains, whereas the CPS design domains addressed in the current work are high-dimensional. Additionally, the examination of methods specifically for the visualization of RL trajectories in design is motivated by the fact that design spaces are known to be complex (e.g., exhibiting fractal self-similarity [11]).

In this work, we specifically identify two aspects that are key for effective interpretation of exploration behavior in a visualization. Firstly, the approach should be able to identify distinct regions of comparable and varying qualities in the design space that are explored by the agent. Secondly, the arrangement of these regions in a 2-dimensional embedding space should be met with a fair accuracy that respects the global structure of the high-dimensional space. These translate to accurately mapping neighbours of a specific region in the design space (local structure) and the accurate spacing of these regions relative to one another (global structure). The

This material is based upon work supported by the Defense Advanced Research Projects Agency through cooperative agreement FA8750-20-C-0002. Any opinions, findings, and conclusions or recommendations expressed in this work are those of the authors and do not necessarily reflect the views of the sponsors.

state-of-the-art embedding algorithms examined here include algorithms that aim at achieving accurate global structure, accurate local structure, or a balance of both.

The remainder of this paper is organized as follows. Section II reviews the methods that are used in this work, namely a set of different embedding algorithms for visualizing the search trajectories of exemplar RL agents. Section III includes examples of the embedding results and discusses the trade-offs between different embedding algorithms. Finally, Section IV concludes the work with a summary of findings, description of limitations, and recommended directions for future work.

II. Methods

In this work, we use RL (see Section II-A) to find solutions to two CPS design problems (see Section II-B) for the purposes of comparing five different embedding algorithms (see Section II-C).

A. Reinforcement-Learning Algorithm

RL algorithms [12] can iteratively learn effective strategies for the sequential decision-making task of exploring the design space. Moreover, they can leverage exploration data in future iterations more efficiently than other design algorithms [13]. This work is based on the exploration trajectories generated by an RL agent from other work [14] that was trained on a proximal policy optimization algorithm [15]. Specifically, the agent state is composed of the design vector, the actions involve tuning the design vector, and the reward is composed of the objectives and constraints.

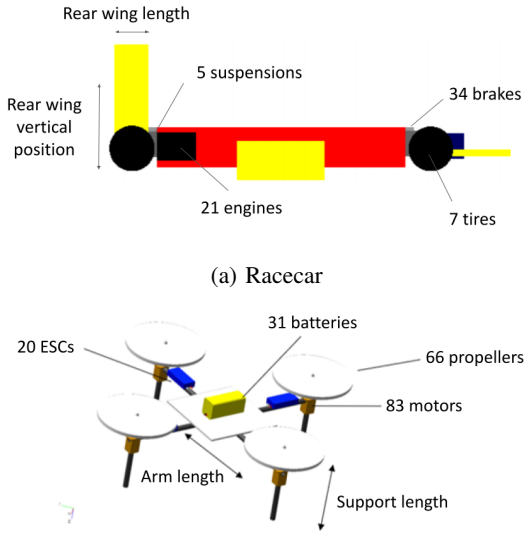
B. Sample CPS Design Problems

1) Aerial Vehicle Design: The aerial vehicle design problem involves the design of a quadcopter with components including batteries, electronic speed controls (ESCs), motors, and propellers. The design space of the problem is comprised of 2 continuous variables – arm length and support length and 4 discrete variables for the choice of batteries, ESCs, motors, and propellers. Figure 1b illustrates an example design with randomly sampled variables. While the number of design variables are low, there are a high number of choices for the discrete variables. The design objective is defined by a vector of 8 sub-objectives to judge the overall performance of the system. These include the mass of the vehicle, hover time and power, distance travelled, and speed, with all 3 of them being measured in 2 scenarios, that of maximum speed and maximum range. To emphasize, these objectives aim at developing lighter, faster, long range and efficient quadcopters. Further, the design is subject to 27 inequality constraints associated with physical interferences in design and the operating limits of the quadcopter components. These include fixed bounds on the 6 design variables, and 15 non-linear inequality constraints.

An RL agent explores the design space of this problem by utilizing a neural network surrogate that was trained using performance data generated from high-fidelity simulations [14]. After training, the policy was evaluated on 6000 seed designs, with each having a trajectory length of 150 steps. We utilize these 6000×150 design vectors to train the embeddings.

2) Racecar Design: The racecar design problem is based on prior work on SAE Formula vehicles [16], [17]. It involves multiple sub-systems of the vehicle such as suspension, engine, and others. Figure1a shows a schematic of the vehicle sub-systems with some of the design variables. The design space of the problem is comprised of 29 continuous (e.g., cabin length, wing length) and 10 discrete (e.g., engine choice from a set of 21 engines) variables. By considering all possible discrete values and merely 10 values for the continuous variables, the size of the combinatorial space is of the order of 10^{39}, a value comparable to the state space size of 10^{40} in chess [18]. The design objective is defined by a set of 11 sub-objectives to judge the overall performance of the system. These are the mass of the vehicle, center of gravity height, drag, downforce, acceleration, crash force, impact attenuator volume, cornering velocity, braking distance, suspension acceleration, and pitch moment. Further, the design is subject to 80 practical and natural inequality constraints. These include fixed or linearly dependent bounds on the 39 design variables, and 2 non-linear inequality constraints.

An RL agent explores the design space of this problem by utilizing a neural network surrogate that was trained using performance data generated from analytical expressions [14]. After training, the policy was evaluated was evaluated on 6000 seed designs, with each having a trajectory length of 20 steps. We utilize these 6000×20 design vectors to train the embeddings.



(a) Racecar

(b) Aerial vehicle

Fig. 1: Example CPS design problems.

C. Embedding Algorithms

1) PCA: Principal Component Analysis (PCA) is a linear dimensionality reduction technique that preserves the global

structure of the data by rotation of the axes such that each successive axis captures as much variance as possible [19]. This makes it suitable to visualize exploration trajectories in a reduced space (for instance [13]). Further, as the algorithm is based on linear transformations, it is computationally inexpensive. However, it does not identify distinct regions in the design space as it transforms the entire data set without any local information. This limits its applicability for the interpretation of search patterns in the design space. However, a PCA-based visualization can serve as a benchmark for evaluating the global structure of algorithms that attempt to achieve a good local structure at the expense of global structure.

2) t-SNE: The t-distributed stochastic neighbourhood embedding (t-SNE) is a non-linear stochastic dimensionality reduction technique that preserves the neighborhood of a point from a high-dimensional space to an embedding space of lower dimensionality [20]. Specifically, it minimizes the Kullback–Leibler divergence of the conditional distributions of pairwise similarities of these spaces. This results in the emergence of local clusters of data in the embedding space. While this makes it suitable for identifying distinct regions explored by a design agent, it functions at the expense of losing the global structure of exploration data. As the relative locations of various clusters is not mapped accurately, it is less suitable for visualizing trajectories that traverse various regions in the design space. Lastly, this approach is more computationally expensive than PCA and involves tuning several hyper parameters including perplexity, learning rate, and number of steps.

3) UMAP: Uniform Manifold Approximation and Projection (UMAP) is a general framework for manifold learning and dimensionality reduction that finds a fuzzy topology structure in the low-dimensional space that is similar to the high-dimensional space, with the difference being measured by cross-entropy [21]. Specifically, it constructs a weighted graph of nearest neighbours and iteratively moves points away from or close to each other in the embedding space based on the high-dimensional space. Similar to t-SNE, this algorithm is effective in identifying distinct regions in the design space but fails to preserve global structure. In addition, UMAP is faster than t-SNE but significantly more sensitive to initialization. It also still involves tuning hyperparameters including the number of nearest neighbors and minimum distance between points in the embedding space.

4) TriMap: TriMap is a dimension reduction technique that preserves the global structure of the data by utilizing triplet constraints [22]. As this approach attempts to preserve both local and global structure, we identify TriMap as a potential candidate for visualizing design space exploration. However, its global structure preserving capability is known to be heavily dependent on the initialization [23].

5) PaCMAP: Pairwise Controlled Manifold Approximation Projection (PaCMAP) is a dimension reduction technique that works well for preserving both the local and global structure of the dataset [23]. Specifically, this is achieved by utilizing three kinds of pairs: neighbour pairs, mid-near pairs, and further pairs. It is known to perform better than UMAP on local structure preservation and better than TriMAP for global structure preservation [23]. In the context of exploration, local information will keep consecutive points on a trajectory close-by when searching within a specific region and global information will help map distant points on the trajectory in different regions. Moreover, global information also helps map points on several nearby and far away trajectories relative to each other. Lastly, PaCMAP is less dependent on initialization than other algorithms and requires less hyperparameter tuning.

D. Basis for Comparison

Once an embedding algorithm is trained, two representative RL agent trajectories are plotted on the embedding. Several metrics are then used for comparison between embeddings.

1) **Training time**: The time used to train. All time comparisons were performed in a single Google Colab instance with an Intel(R) Xeon(R) CPU @ 2.20GHz with 25 GB RAM.
2) **Correlation of pairwise distances**: This measure quantifies the accuracy with which the global structure of the high-dimensional design space is captured. Specifically, 1 million pairs of design vectors are randomly sampled from the exploration dataset and the Pearson correlation coefficient of the Euclidean distances between the high-dimensional and embedding spaces are calculated.
3) **Identification of distinct regions**: This is a qualitative evaluation that assesses whether or not the trained embedding shows distinct regions of self-similar designs. The authors identified the regions and there was agreement among them.
4) **Interpretability of search**: This summarizes the effectiveness of capturing local and global structure, and thereby the interpretability of the RL trajectories. The authors evaluated the interpretability and there was agreement among them.

III. Results and Discussion

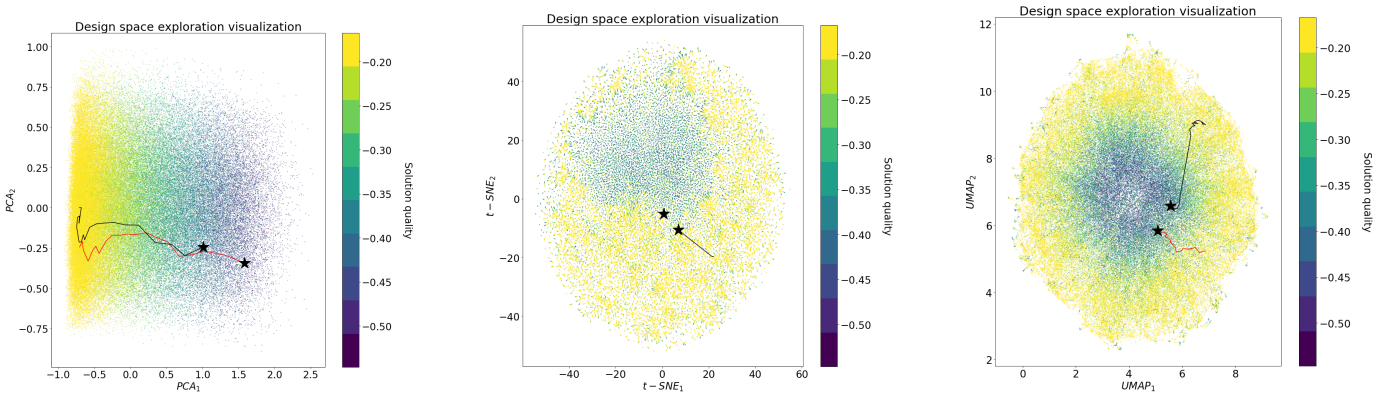
The embeddings for the racecar design problem, as well as representative RL trajectories, are shown in Figure 2. Table I summarizes the results for the racecar design problem.

The character of this design problem is such that three levels of design quality immediately become apparent. These are identified separately as distinct regions in PaCMAP (see Figure 2e), while in all other embeddings they are only distinguished by color. This showcases the effectiveness of PaCMAP in identifying distinct regions in the design space.

However, Table I shows that the correlation values (reflecting preservation of global structure) are the highest for PCA. This value is moderate for PaCMAP, and the other algorithms have poor values. This is also reflected visually in the relative locations of the trajectories in Figure 2. While PCA does not reflect any local structure, it can serve as a visual benchmark for evaluating the global structure obtained by other algorithms. The RL agents consistently proceed from areas of low quality towards areas of higher quality in all the

TABLE I: Summary of comparison between different embedding algorithms for the racecar example problem

Embedding Algorithm	Training Time	Correlation	Distinct Regions	Interpretability
PCA	1.2 seconds	0.82	No	Fair as no distinct regions of varying quality are identified, but very accurate relative locations of trajectory (as reflected by correlation)
t-SNE	23 minutes	0.04	Yes	Poor. Although distinct regions are identified, the trajectories do not respect the global structure.
UMAP	3.1 minutes	-0.114	Yes	Poor. Although distinct regions are identified, the trajectories do not respect the global structure.
TriMap	2.4 minutes	0.18	Yes	Fair. Distinct regions are identified. The 2 trajectories somewhat respect the PCA trajectories. However, correlation is much lower than PaCMAP, so in general, trajectories will be less accurate.
PaCMAP	3 minutes	0.56	Yes	Good. Distinct regions are clearly defined. Trajectories respect global structure as in PCA.



(a) PCA (b) t-SNE (c) UMAP

(d) TriMap (e) PaCMAP

Fig. 2: Embeddings trained on the racecar design problem with two example trajectories shown.

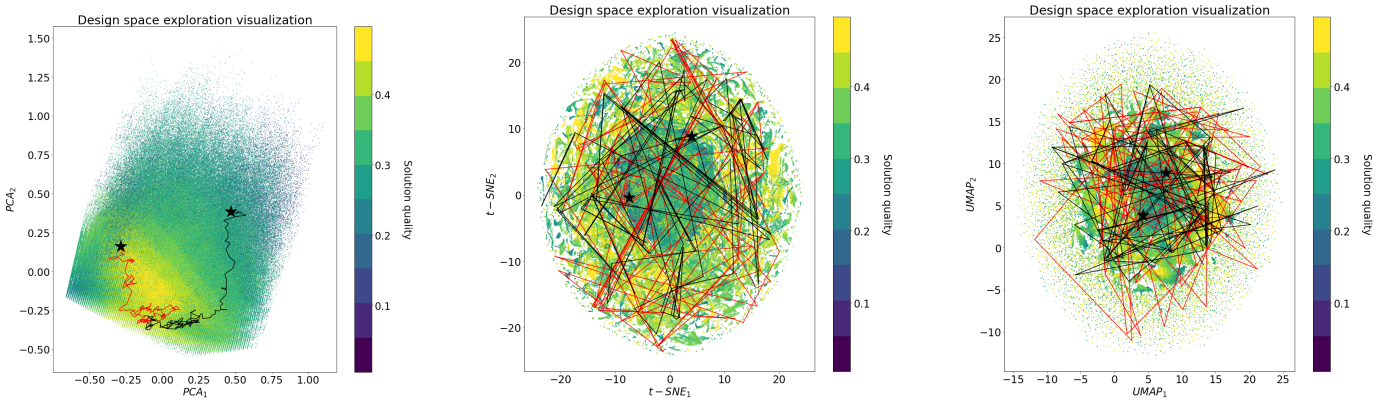
embeddings. Because of the relatively short trajectories used in the racecar design problem (merely 20 iterations for the trained policy), the trajectories tend to appear relatively smooth on all embeddings. Lastly, the computation time required to learn the embeddings is highest for t-SNE, followed by comparable values for UMAP, TriMAP and PaCMAP. The value for PCA is much lower than other algorithms since it is a linear transformation.

The embeddings for the aerial vehicle design problem, as well as representative RL trajectories, are shown in Figure 3. Table II summarizes the results for the aerial vehicle design problem.

In this second design problem, the differences between several of the methods are exacerbated. Firstly, distinct regions are only clearly observed in the PaCMAP embedding. While the t-SNE and UMAP embeddings (Figures 3b and 3c, respectively) identify some distinct regions, they offer extremely noisy RL agent trajectories as they poorly capture the global structure of the data. This contrasts with the smoother trajectories provided by TriMap and PaCMAP (Figures 3d and 3e) and especially the smoothness of PCA (Figure 3a). More importantly, the TriMAP and PacMAP respect the relative

TABLE II: Summary of comparison between different embedding algorithms for the aerial vehicle example problem

Embedding Algorithm	Training Time	Correlation	Distinct Regions	Interpretability
PCA	1.7 second	0.82	No	Fair as no distinct high quality regions but very accuarte relative locations of trajectory (as reflected by correlation)
t-SNE	5.6 hours	-0.13	Yes	Poor. Although distinct regions are identified, they are not arranged well. Thus trajectories are not interpretable.
UMAP	1.8 hours	-0.17	Yes	Poor. Although distinct regions are identified, they are not arranged well. Thus trajectories are not interpretable.
TriMap	46 minutes	0.41	No	Poor. Although relative locations of trajectory to each other are somewhat aligned with PCA trajectories, distinct regions are not identified.
PaCMAP	1.4 hours	0.42	Yes	Good. Distinct regions are clearly defined. The relative locations of trajectories are aligned, but with some warping.



(a) PCA (b) t-SNE (c) UMAP

(d) TriMap (e) PaCMAP

Fig. 3: Embeddings trained on the aerial vehicle design problem with two example trajectories shown.

location of trajectories to a fair extent when compared with PCA. This is also reflected in the moderate values of the pairwise distance correlations.

It should be noted that the aerial vehicle design problem requires embeddings to be trained on 7.5× more data than the racecar design problem. This consequently increased the training time of most methods by orders of magnitude (see Table I). However, the PCA embedding time was still measured in seconds, a testament to the scalability of the simpler approach.

IV. Conclusion and Recommendations

This work is motivated by the emerging need not only to create more effective design algorithms, but to better understand the search behaviors of those algorithms. This is important to both gain a more nuanced understanding of the algorithms themselves to allow for iterative refinement,

but also to expose information that is helpful for supporting human-AI collaboration.

Results were consistent across the two design problems used here, despite the differences between the problems themselves. The most useful visualizations were consistently provided by the PCA and PaCMAP embeddings. If training time is a significant factor, then PCA is preferable. If the ability to resolve subgroups of self-similar designs is desirable, then PaCMAP should be chosen instead.

This work represents an initial exploration into this application of data visualization, and is limited in some respect. First, we only examine the embeddings on two sample design problems. However, design problems are known to be highly variable in their characteristics. Future work should therefore perform a more detailed comparison on a wider set of design problems. Second, this work only investigates the visualization of trajectories generated by the RL agent, which is an iterative point-based algorithm. Future work should and consider the additional challenges posed by set-based design algorithms such as genetic algorithms, particle swarm optimization, and others.

ACKNOWLEDGMENT

The authors are grateful to the Southwest Research Institute for providing simulation capabilities for the drone case study used in this work. We are also grateful to Susmit Jha and Adam Cobb of SRI International for their feedback on early versions of the RL agents used in this work.

REFERENCES

[1] S. A. Seshia, S. Hu, W. Li, and Q. Zhu, "Design automation of cyber-physical systems: Challenges, advances, and opportunities," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 9, pp. 1421–1434, 2017.

[2] A. Agrawal, S. J. Won, T. Sharma, M. Deshpande, and C. McComb, "A multi-agent reinforcement learning framework for intelligent manufacturing with autonomous mobile robots," *Proceedings of the Design Society*, vol. 1, pp. 161–170, 2021.

[3] H. Neema, Z. Lattmann, P. Meijer, J. Klingler, S. Neema, T. Bapty, J. Sztipanovits, and G. Karsai, "Design space exploration and manipulation for cyber physical systems," in *IFIP First International Workshop on Design Space Exploration of Cyber-Physical Systems (IDEAL'2014), Springer-Verlag Berlin Heidelberg*, 2014, p. 8.

[4] J. T. Gyory, N. F. Soria Zurita, J. Martin, C. Balon, C. McComb, K. Kotovsky, and J. Cagan, "Human versus artificial intelligence: A data-driven approach to real-time process management during complex engineering design," *Journal of Mechanical Design*, vol. 144, no. 2, 2022.

[5] B. Song, J. Gyory, G. Zhang, N. F. Soria Zurita, G. Stump, J. Martin, S. Miller, C. Balon, M. Yukish, C. McComb, and J. Cagan, "Decoding the agility of artificial intelligence-assisted human design teams," *Design Studies*, 2022.

[6] C. Molnar, G. Casalicchio, and B. Bischl, "Interpretable machine learning–a brief history, state-of-the-art and challenges," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2020, pp. 417–431.

[7] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[8] S. Deshpande and J. Schneider, "Vizarel: A system to help better understand rl agents," *arXiv preprint arXiv:2007.05577*, 2020.

[9] J. Wang, L. Gou, H.-W. Shen, and H. Yang, "Dqnviz: A visual analytics approach to understand deep q-networks," *IEEE transactions on visualization and computer graphics*, vol. 25, no. 1, pp. 288–298, 2018.

[10] N. Douglas, D. Yim, B. Kartal, P. Hernandez-Leal, F. Maurer, and M. E. Taylor, "Towers of saliency: A reinforcement learning visualization using immersive environments," in *Proceedings of the 2019 acm international conference on interactive surfaces and spaces*, 2019, pp. 339–342.

[11] C. McComb, J. Cagan, and K. Kotovsky, "Optimizing design teams based on problem properties: computational team simulations and an applied empirical test," *Journal of Mechanical Design*, vol. 139, no. 4, p. 041101, 2017.

[12] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[13] X. Y. Lee, A. Balu, D. Stoecklein, B. Ganapathysubramanian, and S. Sarkar, "A case study of deep reinforcement learning for engineering design: Application to microfluidic devices for flow sculpting," *Journal of Mechanical Design*, vol. 141, no. 11, 2019.

[14] A. Agrawal and C. McComb, "Reinforcement learning with variable fidelity models for design of cyber-physical systems," in *Submitted to ASME 2022 International Design Engineering Technical Conferences Computers and Information in Engineering Conference*, 2022.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[16] N. F. Soria Zurita, M. K. Colby, I. Y. Tumer, C. Hoyle, and K. Tumer, "Design of complex engineered systems using multi-agent coordination," *Journal of Computing and Information Science in Engineering*, vol. 18, no. 1, 2018.

[17] S. Lapp, K. Jablokow, and C. McComb, "Kaboom: an agent-based model for simulating cognitive style in team problem solving," *Design Science*, vol. 5, 2019.

[18] S. Steinerberger, "On the number of positions in chess without promotion," *International Journal of Game Theory*, vol. 44, no. 3, pp. 761–767, 2015.

[19] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, vol. 2, no. 11, pp. 559–572, 1901.

[20] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[21] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[22] E. Amid and M. K. Warmuth, "Trimap: Large-scale dimensionality reduction using triplets," *arXiv preprint arXiv:1910.00204*, 2019.

[23] Y. Wang, H. Huang, C. Rudin, and Y. Shaposhnik, "Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization," *Journal of Machine Learning Research*, vol. 22, pp. 1–73, 2021.