DATA-DRIVEN RESEARCH ON ENGINEERING DESIGN THINKING AND BEHAVIORS IN COMPUTER-AIDED SYSTEMS DESIGN: ANALYSIS, MODELING, AND PREDICTION

DATA-DRIVEN RESEARCH ON ENGINEERING DESIGN THINKING AND BEHAVIORS IN COMPUTER-AIDED SYSTEMS DESIGN: ANALYSIS, MODELING, AND PREDICTION

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Mechanical Engineering

by

Molla Hafizur Rahman
Bangladesh University of Engineering and Technology, 2015
Bachelor of Science in Naval Architecture and Marine Engineering

August 2022
University of Arkansas

This thesis is approved for recommendation to the Graduate Council

---

Zhenghui Sha, Ph.D.
Dissertation Director

David Jensen, Ph.D. Committee member	Uche Wejinya, Ph.D. Committee member
Darya Zabelina, Ph.D. Committee member	Lu Zhang, Ph.D. Committee member

Abstract

Research on design thinking and design decision-making is vital for discovering and utilizing beneficial design patterns, strategies, and heuristics of human designers in solving engineering design problems. It is also essential for the development of new algorithms embedded with human intelligence and can facilitate human-computer interactions. However, modeling design thinking is challenging because it takes place in the designer's mind, which is intricate, implicit, and tacit. For an in-depth understanding of design thinking, fine-grained design behavioral data are important because they are the critical link in studying the relationship between design thinking, design decisions, design actions, and design performance. Therefore, the research in my dissertation aims to develop a new research platform and new research approaches to enable fine-grained data-driven methodology that helps foundationally understand the designers' thinking and decision-making strategies in engineering design. To achieve this goal, my research has focused on modeling, analysis, and prediction of design thinking and designers' sequential decision-making behaviors.

In the modeling work, different design behaviors, including design action preferences, one-step sequential decision behavior, contextual behavior, long short-term memory behavior, and reflective thinking behavior, are characterized and computationally modeled using statistical and machine learning techniques. For example, to model designers' sequential decision-making, a novel approach is developed by integrating the Function-Behavior-Structure (FBS) design process model into deep learning methods, e.g., the long short-term memory (LSTM) model and the gated recurrent unit (GRU) model.

In the work on analysis, this dissertation focuses primarily on different clustering analysis techniques. Based on the behaviors modeled, designers showing similar behavioral patterns can be clustered, from which the common design patterns can be identified. Another analysis performed in this dissertation is on the comparative study of different sequential learning techniques, e.g., deep learning models versus Markov chain models, in modeling sequential decision-making behaviors of human designers. This study compares the prediction accuracy of different models and helps us obtain a better understanding of the performance of deep-learning models in modeling sequential design decisions.

Finally, in the work related to prediction, this dissertation aims to predict sequential design decisions and actions. We first test the model that integrates the FBS model with various deep-learning models for the prediction and evaluate the performance of the model. Then, to improve the accuracy of the prediction, we develop two approaches that directly and

indirectly combine designer-related attributes (static data) and designers' action sequences (dynamic data) within the deep learning-based framework. The results show that with appropriate configurations, the deep-learning model with both static data and dynamic data outperforms the models that only rely on the design action sequence. Finally, I developed an artificial design agent using reinforcement learning with a data-driven reward mechanism based on the Markov chain model to mimic human design behavior. The model also helps validate the hypothesis that the design knowledge learned by the agent from one design problem is transferable to new design problems.

To support fine-grained design behavioral data collection and validate the proposed approaches, we develop a computer-aided design (CAD)-based research platform in the application context of renewable engineering systems design. Data are collected through three design case studies, i.e., a solarized home design problem, a solarized parking lot design problem, and a design challenge on solarizing the University of Arkansas (UARK) campus. The contribution of this dissertation can be summarized in the following aspects. First, a novel research platform is developed that can collect fine-grained design behavior data in support of design thinking research. Second, new research approaches are developed to characterize design behaviors from multiple dimensions in a latent space of design thinking. We refer to such a latent representation of design thinking as design embedding. Furthermore, using deep learning techniques, several different predictive models are developed that can successfully predict human sequential design decisions with prediction accuracy higher than traditional sequential learning models. Third, by analyzing designers' one-step sequential design behaviors, common and beneficial design patterns are identified. These patterns are found to exist in many high-performing designers in the three respective design problems studied. Fourth, new knowledge has been obtained on the ability of deep learning-based models versus traditional sequential learning models to predict sequential design decisions of human designers. Finally, a novel research approach is developed that helps test the hypothesis of transferability of design knowledge. In general, this dissertation creates a new avenue for investigating designers' thinking and decision-making behaviors in systems design context based on the data collected from a CAD environment and tested the capability of various deep-learning algorithms in predicting human sequential design decisions.

ACKNOWLEDGEMENTS

First and foremost, thanks to the ALMIGHTY ALLAH for HIS showers of blessings throughout my Ph.D. research and life. I would like to express my deepest gratitude to my advisor, Dr. Zhenghui Sha, who has been pivotal throughout my PhD journey by providing invaluable guidance. Without his constant support and feedback, this dissertation would not have been completed. His critical thinking, vision, motivation, and work ethic deeply inspired me during my Ph.D. career and beyond. I would like to also thank Dr. David Jensen for his strong support during my Ph.D. I really enjoyed the weekly presentations on different topics by System Integration & Design Informatics (SIDI) lab members, including Laxmi Poudel, Xingang Li, Yinshuang Xiao, Jared Poe, John Clay, and Saivipulteja Elagandula, and Complex Adaptive Engineered Systems Research (CAESR) lab members, including Charlie DeStefano, Jonathan Ashley, and Marvin Arroyo. I appreciate their constructive feedback on my research.

I feel very fortunate to have the opportunity to work with talented people like Kaleb Porter, Luke Godfrey, Dr. Corey Schimpf, Dr. Charles Xie, Dr. Michael Gashler, Dr. Shuhan Yuan, and Dr. Alparslan Emrah Bayrak. Without their help and suggestions, this research could not have been accomplished. I am grateful to John Clay, Xingang Li, Rebecca White, and Dr. Darya Zabelina who have helped me a lot in the design of human-subject experiments and data collection.

I am indebted to my father: Molla Habibur Rahman, and my mother: Parvin Akther, for their immense support, prayers, care, and sacrifices in educating me and preparing me for handling tough times. I would also extend my thanks to my brother, nephew and friends with whom I have spent time during stressful times.

I want to express my deepest appreciation to the staff of the Department of Mechanical Engineering at the University of Arkansas for answering all of the queries and providing assistance to me from time to time. I would like to acknowledge the financial support provided by the National Science Foundation (NSF) and the University of Arkansas for my Ph.D. dissertation research. Finally, I would like to thank my committee members, Dr. David Jensen, Dr. Uche Wejinya, Dr. Darya Zabelina and Dr. Lu Zhang for evaluating my Ph.D. work and providing me with constructive feedback throughout my Ph.D. proposal and the final defense of this dissertation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Background and Motivation

Design is a purposeful activity that aims to meet a set of requirements for an artifact [3, 4]. It typically involves defining problems and solving them [5]. The former stage usually transforms design from ill-defined problems to well-defined ones, from which both design variables and constraints can be identified and the design space is determined. This step often requires design ideation, conceptualization, and requirement analysis. The later stage relies on different strategies of searching to find the most appropriate solutions within the identified design space. In both stages, the design is highly dependent on the knowledge of a designer consisting not only of explicit knowledge that can be obtained from encoded information, such as text and drawings, but also tacit knowledge that is difficult to visualize and mainly accumulated from experience [6]. In addition, designers are bounded rational due to human limited information processing capability [7]. In this process, engineering design

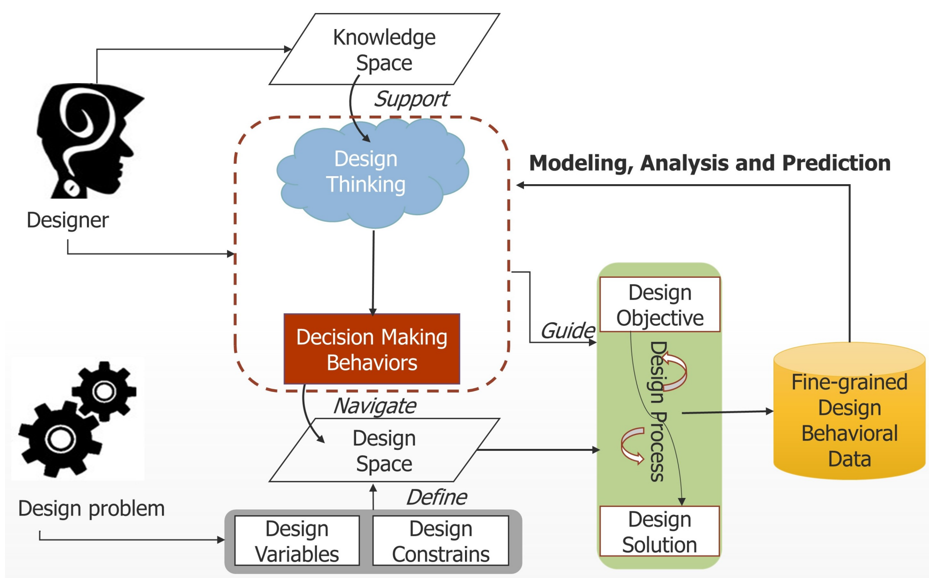


Figure 1.1: Engineering design thinking (EDT) and decision-making in the design process. EDT is a complex process of inquiry and learning that designers perform in a system context, making decisions as they proceed and often working collaboratively.

thinking (EDT), as shown in Figure 1.1, plays an important role in the bridge between design knowledge and design problems, given the bounded rationality of designers, to guide designers' operations to navigate through the design space in a stepwise, sequential but iteratively (a.k.a. sequential decision-making) in order to achieve the design objective. In the context of engineering design, design thinking refers primarily to iterations of exploration (i.e. divergent thinking) and exploitation (i.e., convergent thinking) in the search for design solutions [8]. More generally speaking, design thinking is the cognitive activities of designers during a design process. Their decision-making strategies in the design process are guided by their design thinking, and their corresponding actions are reflected through the design task. Design thinking guides how designers apply design principles to generate, evaluate and represent concepts to meet the stated goals [9, 10].

Designers' sequential decision-making involves the selection of design actions and determination of design parameters so that they can get the best output. These sequential decision-making strategies are often the key features that differentiate expert designers from novices. Therefore, these strategies compose the essence of human intelligence in design. To fundamentally understand designers' sequential decision-making strategies, two essential elements of a design process must be considered: the design problem (i.e., the artifact to be designed) and the designer [5]. As mentioned above, a well-defined design problem typically outlines the design space where designers search for solutions. The search is realized by actions/operations taken on the design artifact, such as adding/deleting a component and modifying the values of the corresponding design variables. A designer, on the other hand, is the entity who actually makes design decisions guided by his/her own design thinking [11] that are often influenced by prior knowledge and experience. Based on the response to the present action taken (e.g., whether adding a component violates the design constraints or not), as well as the present resources (e.g., current budget), a designer will make the next decision to navigate through the design space searching for acceptable solutions. Therefore, to successfully model designers' sequential design decisions, two classes of information must be acquired: the sequential actions (a decision is a commitment to an action [12]) taken by a designer and the designer-related attributes that often form his/her design thinking.

Scientific investigation of designers' thinking and sequential decision-making is of great interest. For example, in design problems with large design spaces or designs accompanied by significant uncertainties, trade-off strategies, and decisions on when and where to explore or exploit the design space are important to the quality of design outcomes and the resources needed to achieve such outcomes. Therefore, an in-depth understanding of designers' thinking and decision-making behaviors is critical to the discovery of beneficial design heuristics that can, in turn, be used to facilitate the design process and improve design automation. In addition, the discovery of effective decision-making strategies from expert designers can help train novice designers. Moreover, computational modeling of decision-making behaviors and design thinking can help build artificial design agents that can be coded into computer-aided design (CAD) systems to assist human designers and facilitate human-computer interactions in design.

However, studying designers' thinking and their decision-making behaviors is challenging. This is because human design thinking and decisions are the result of mental processes that are hidden, implicit, and intricate. Therefore, the research conducted in this area has been primarily empirical and data-driven. However, several data requirements are essential for design thinking research. First, complex design is subdivided into several design stages, and design thinking plays a vital role in organizing and iterating through these stages. Therefore, data that cycle through all these design stages and capture all the design variables are required. Second, temporal granularity is critical because how frequent designers' actions are collected may reflect the nuance in different design strategies adopted by different designers. Therefore, fine-grained behavioral data are preferred in design thinking research. Third, in complex design, the design process takes place over a long period of time, and design decisions are highly correlated with multiple interdependent variables. So, the design patterns generated from these design decisions could be weak, and the strategies can vary between different designers. Therefore, it is difficult to capture prominent design patterns from the designers' behavioral data. Fourth, designers' attributes (i.e. demographics), which are related to designers' knowledge, are not always available due to privacy issues. Even if the data containing designers' attributes are available, it is not explicit which design attributes are useful to understand design thinking and decision making. Finally, data collection for studying design thinking and decision making requires a long time, and sometimes designers are not readily available (e.g., during pandemic situation). So, it is often challenging to collect a sufficient amount of data for model training. The data scarcity issue could become even more serious if the design problems used for data collection are new because there would not be many designers working on those problems yet.

To address these challenges, this dissertation conducted research in several directions. First, a set of data requirements that are essential to conduct and validate design thinking research is identified. For example, the data should cover both intra-stage and inter-stage design iterations; should be high-fidelity and non-intrusiveness; should capture designers' rational behavior; and should be collected in multiple formats, such as texts, images, and videos. More details on these requirements are described in Chapter 4. Based on these requirements, we develop a CAD-based research platform that can help collect fine-grained design behavioral data in systems design context that meet the data requirements discussed above. Second, we develop a computational model to predict human sequential design decisions. The uniqueness of the model is that it integrates a design process model and deep learning techniques that capture both short-term and long-term design patterns of design behaviors. Thus, the developed model is able to address the challenge of identifying and extracting complex design patterns. Third, to solve the availability of the designer-related attributes issue, we develop a clustering technique that can aggregate designers' attributes directly from designers' sequential design actions. Then, a novel deep learning-based approach is developed that can combine the aggregated designer-related data and their sequential design action data to predict human sequential design decisions. Finally, to solve the data scarcity issue, we develop a framework based on reinforcement learning to test the transferability of design knowledge across different design problem contexts. In this framework, we first develop a design agent to mimic human design behavior. The design knowledge learned by the reinforcement learning agent from the source problem is then transferred to the target design problem. As the agent is trained by the data set for the source design problem, it does not require additional training data for the target design problem. Thus, it solves the challenge regarding data scarcity.

1.2 Research Hypothesis, Questions and Objective

The main objective of this dissertation research is to develop data-driven computational approaches to modeling design behaviors in engineering systems design, particularly in a computer-aided design (CAD) environment, to understand engineering design thinking and predict future design decisions. The central hypothesis is that data mining and machine learning of CAD action logs can help model and identify design thinking and decision-making as well as the relationship between them. To achieve the objective of the research and test the central hypothesis, two research questions are identified, which drive data-driven research and scientific inquiries into modeling, analysis, and prediction of engineering design thinking and behaviors in computer-aided systems design. These questions are:

1. **RQ1:** What are the relationships between design behaviors and design outcomes?

2. **RQ2:** How to predict designers' sequential decisions in computer-aided design based on the characterization of different aspects of design thinking?

Table 1.1 shows the overview of our proposed research objective, the central research hypotheses, and the research approaches. Furthermore, Table 1.2 provides an overview of the individual research question, the hypothesis, and the dissertation chapters that correspond to the research questions. Additionally, Table 1.2 contains the expected research outcomes of the individual research question.

1.3 Research Approach

To answer the two RQs, we proposed several research approaches, as shown in Table 1.2. Each of the research approaches is discussed below:

1. *Integrate the Function-Behavior-Structure (FBS) design process model and the Markov chain model to automatically cluster and extract sequential behavioral patterns.*

   Based on this approach, we model sequential one-step behavior and, by analyzing it, we identify common and beneficial design patterns. Thus, this approach helps to fulfill the research objective of modeling and analyzing design behaviors. Additionally, it lays the foundation for modeling design behaviors in multiple different dimensions in addition to the one-step sequential behavior, thereby it helps answer RQ1.

2. *Use distribution analysis, machine learning and deep learning techniques to model and cluster design behaviors and study their relationship to design performance.*

   Based on the second approach, we identify different design behaviors from five dimensions, i.e.,design action preference, one-step sequential decision behavior, contextual behavior (i.e., the behavioral pattern in a context of several adjacent design actions), long-step sequential behavior and reflective thinking behavior. Then, we computationally model those behaviors using different statistical and machine learning techniques. In particular, the design action distribution is used to model the design action preference. The first-order Markov chain model is used to model one-step sequential behavior. Doc2Vec (an natual language processing (NLP) tool for representing documents

Table 1.1: The research objective, the central hypothesis, and the proposed research approaches

Research objective	To develop data-driven computational approaches to modeling design behaviors in engineering systems design, particularly in computer-aided design (CAD) environment, for understanding engineering design thinking and predicting future design decisions.
Central hypothesis	Data mining and machine learning of CAD action logs can help model and identify design thinking and decision-making as well as the relationship between them.
Research approaches	The approaches to achieving the central research objective and to testing the central research hypothesis are as follows: 1. Integrating the Function- Behavior-Structure (FBS) design process model and Markov chain model to automatically cluster and extract sequential behavioral patterns. 2. Using distribution analysis, machine learning and deep learning techniques to model and cluster design behaviors and study their relationship to design performance.. 3. Utilizing the FBS design process model and long short-term memory (LSTM) model to predict sequential design decisions. 4. Developing a deep-learning based approach to predicting design decisions combining designer-related attributes and sequential design actions. 5. Developing a reinforcement learning-based framework to test the transferability of design knowledge from one design problem to other design problems.

(since design actions are stored as text data) as a vector) is used to model contextual behavior. The bi-directional long short-term memory unit (LSTM) autoencoder is used to model long-term sequential behavior. And time gap distribution analysis is used to model reflective thinking behavior. Based on the embedding (the latent representation of sequential design action data) obtained from each of these techniques, we cluster designers into different groups. For every generated cluster, we measure the average design quality of the designers in that cluster. Then the clusters of an embedding are compared on the basis of the measured average design quality. The relationship discovered between design behaviors and design quality from this approach helps answer RQ1.

3. *Utilize the FBS design process model and long short-term memory (LSTM) model to predict sequential design decisions.*

   From this approach, we understand how designers' short-term and long-term memory play a role in their sequential design decision making. Additionally, we compare deep learning models (e.g., the LSTM model and the gated recurrent unit (GRU) model) with traditionally used sequential learning models, such as Markov chain models. From this comparative study, we understand which models perform better in predicting the sequential design decisions of human designers. This approach helps to achieve the research objective by answering RQ2.

4. *Develop a deep learning-based approach to predicting design decisions combining designer-related attributes (static data) and sequential design actions (dynamic data).*

   In this approach, two different combination methods, namely the direct method and the indirect method, are developed. In the direct method, designers' attributes are directly passed into the deep learning model, while in the indirect model, designers' attributes are first separately processed. The processed information is then added indirectly to the deep learning model. These methods help understand whether adding designer-related attributes in the deep learning model improves prediction accuracy or not. Therefore, this approach helps answer RQ2.

5. *Develop a reinforcement learning-based framework to test the transferability of design knowledge from one design problem to another design problem.*

This approach first develops a design agent using a reinforcement learning method. Particularly, a data-driven reward mechanism based on the first-order Markov chain is formulated to train the agent. The reinforcement learning algorithm (i.e., the Q-learning model) reinforces the commonly used design patterns. This study focuses on understanding the transferability of design knowledge and uses the learned knowledge from the source problem to predict future design decisions in the target problem. Thus, this approach helps achieve the research objective by answering RQ2.

Table 1.2: The research questions, tasks associated with the research questions and expected outcomes

Research Question 1	**Hypothesis 1**	**Chapters**	**Expected Outcomes**
What are the relationships between design behaviors and design outcomes?	Multi-dimensional characterization of design behaviors helps elicit the relationship between design behavior and design outcomes	Ch.4: Develop an open source research platform for data-driven design thinking research. Ch.5: Use the Markov chain model to model designers' one-step sequential behavior and identify design patterns. Ch.6: Develop approaches to characterize and model design behaviors from multiple dimensions.	• A CAD-based research platform that embodies various unique features for engineering design thinking research. • A general framework that integrates design process models, Markov models, and cluster algorithms to identify sequential design patterns. • A set of beneficial design patterns and heuristics that are commonly used in computer-aided system design practice.

Table 1.2 (Cont.)

Research Question 2	**Hypothesis 2**	**Chapters**	**Expected Outcomes**
How to predict designers' sequential decisions in computer-aided design based on the characterization of different forms of design thinking?	2.1 Integrating designers' attributes into a predictive model improves the prediction accuracy. 2.2 Transferring learned design knowledge from source design problem enable design action prediction for target design problem.	Ch.7: Develop a framework by integrating the design process model and deep learning model for prediction purposes. Ch.8: Extend the framework by incorporating designer-related attributes and psychological attributes. Ch.9: Develop a framework based on reinforcement learning to test the trasferrablity of design knowledge from source design problem to target design problem.	• A deep learning-based framework for predicting designers' sequential design decisions. • A computational framework that combines sequential design actions and designers' attributes to improve predictions of design decisions. • A reinforcement learning-based framework for understanding the transferability of design knowledge learned from the source design problem to the target design problem.

The scope of the research is focused on the design activities made in the CAD processes for ease of data collection, but our approach is generally applicable in any design situation as long as designers' action data can be recorded and collected. Additionally, it is important to mention that the design data is collected mainly in the embodiment design phase with both configuration and parametric design. So, studying certain design research topics, such as design creativity that happens in the early phase (i.e., conceptual design phase), from these data may cause biased results.

1.4 Contributions

The developed approaches are novel and make significant contributions to the field of engineering design. These contributions can be described in three aspects: modeling, analysis, and prediction.

1. **The contributions from the modeling aspect.** From the modeling aspect, the first contribution is that, utilizing the first-order Markov chain model, we devised a novel method to characterize the one-step sequential decision behavior of human designers. Next, based on existing design theories, we identify five design behaviors, including design action preference, one-step sequential behavior, contextual behavior, long-term sequential behavior, and reflective thinking behavior. From the modeled design behavior, a latent representation of design thinking in each of these five dimensions is obtained that is referred to as Design Embedding. This Design Embedding can be further used for other purposes, such as identifying designers with similar behavioral patterns. To model designers' long-term sequential decision-making, we developed an approach that integrates the FBS design process model and the one-hot vectorization to transform design actions to design process stages in order to tackle the high dimensionality associated with the design sequence data and draw insights into design thinking.

2. **The contributions from the analysis aspect.** In the analysis aspect, the contributions can be described as follows. First, the identification of designers with similar behaviors by clustering the modeled behaviors from multiple dimensions is unique and novel. Successful identification of designers with similar behavior has a significant benefit in guiding team-based design. Second, a new metric is developed that can quantify the efficiency of a clustering algorithm in clustering design behaviors with the measurement of Variation of Information, from which the most common design patterns can be identified. Third, we compare the deep learning models with the traditionally used sequential learning models and obtain new knowledge on the capability of deep learning models in predicting designers' sequential design decisions.

3. **The contributions from the prediction aspect.** From the prediction aspect, a major contribution lies in the successful prediction of sequential design decisions using the developed approach that integrates the FBS design process model into a deep learn-

ing framework. Next, the study of combining the attributes of designers (static data) and the sequential design action (dynamic data) provides new knowledge on how well the LSTM model and the GRU model would perform in predicting human sequential design decisions. Additionally, it also provides insight into how well each combination method (i.e., direct vs. indirect) would perform with different deep learning models (i.e., LSTM vs. GRU). To facilitate the generation of static data with limited access to designer-related attributes, e.g., age, gender, etc., we develop a novel technique that automatically identifies designers' aggregated static information using clustering-based techniques. These combination methods eventually improve the prediction accuracy in predicting sequential design decisions. Finally, we gain insight into the predictability of the reinforcement learning agent, where we formulate a novel data-driven reward mechanism using the first-order Markov chain model. The study also corroborates the transferrability of learned design knowledge from the source task to a different target task, and thus enables the prediction of design decisions in the target tasks.

1.5 Outline and Roadmap

The dissertation is mainly divided into two parts. Part 1 focuses on the relationship between design behavior and design outcomes and consists of chapters 4 - 6. Part 2 contains the study of the prediction of design decisions and consists of Chapters 7-9. The dissertation road map is provided at the end of this chapter.

Chapter 2. Literature Review discusses the relevant research on methods and tools to study design thinking research in the engineering design domain. Then the chapter discusses how design thinking study can be conducted using computer-aided design software and some of the attempts that use CAD software and self-developed tools for non-intrusive data logging. The chapter then discusses different studies on sequential decision-making strategies. In addition, approaches to identifying and predicting sequential design decisions are discussed. Recently, deep learning has been successful in many areas, including design field. So, the focus has been shifted to the research on deep learning methods in design research that study different problems, including design optimization, design ideation, and design behavior. Finally, we conclude the chapter with identified research gaps from the relevant literature.

Chapter 3. Design Experiment and Data Collection focuses on design experiments that have been carried out to collect data sets for my dissertation research. Also, this chapter discusses the goals, corresponding tasks, and outcomes of each of the design experiments according to their design contexts. A detailed data collection procedure is then described from these experiments. In addition, this chapter discusses the type of data, the data cleaning process, and how the design process model can be used to convert the design action sequence into a sequence of data from the design process stages.

Chapter 4. Open Source Design Platform for Data-Driven Design Thinking Research first identifies and discusses the essential data requirements for engineering design thinking research. Then it presents the open-source design research platform (i.e., Energy3D) and its different features, which includes data collection, features for engineering system design, and features for design experiments.

Chapter 5. Modeling One-Step Sequential Behavior Using Markov Chain Model provides a novel method to model one-step sequential behavior using the first-order Markov chain model. Also, this chapter introduces three different clustering methods, including K-mean clustering, hierarchical clustering, and network-based clustering method, that can cluster designers' sequential design behaviors. An evaluation method is also presented to validate the clustering results. Finally, in this chapter, we discuss several common design patterns identified from the clustering study.

Chapter 6. Modeling Design Behavior from Multiple Dimension first focuses on identifying important design behaviors from the existing design theory. Particularly, this chapter presents five design behaviors which include design action preference, one-step sequential behavior, contextual behavior, long-term sequential behavior, and reflective thinking behavior. Then it presents the computational methods for characterizing each of the design behaviors. The latent representation of each design behavior is referred to as design embedding. Based on the design embedding, designers are then clustered into different groups. Finally, based on the clustering results, the performance of the designers of different clusters is compared and then useful design patterns are identified.

Chapter 7. Predicting Design Decision Using the Deep Learning Method describes a deep learning-based framework that integrates a design process model for the prediction of future design actions. We train and test the model with a K-fold cross-validation approach. For evaluation purposes, in addition to prediction accuracy, we use the recall, precision, F1, and area under the receiver operating characteristic curve (AUROC score) to compare with traditionally used sequential models (i.e., Markov Chain model, hidden

Markov model) and baseline models (i.e., random model and repetitive).

Chapter 8 Combining Static and Dynamic Data for Predicting Sequential Decision Making presents a novel deep learning model that leverages designers' action data as well as designers' attributes for future action prediction. We develop two different combination methods where we use clustering methods to identify designers' attributes.

Chapter 9 Developing Reinforcement Learning-Based Framework for Understanding the Transferrability of Design Knowledge introduces a design agent based on a reinforcement learning model that uses a novel data-driven reward mechanism based on the first-order Markov chain. Then it shows the ability of the model in transferring the learned design knowledge from the source design problem to the target design problem.

Chapter 10 Conclusion and Future Work provides a concluding remark and the major contributions of this dissertation. We also revisit the research questions presented at the beginning of the dissertation and answer those questions based on the research conducted. Furthermore, this chapter discusses several directions of future studies that include possible use cases of design embedding, methods to add psychological measures to the developed deep learning models, and approaches to enabling human-AI collaboration in CAD software.

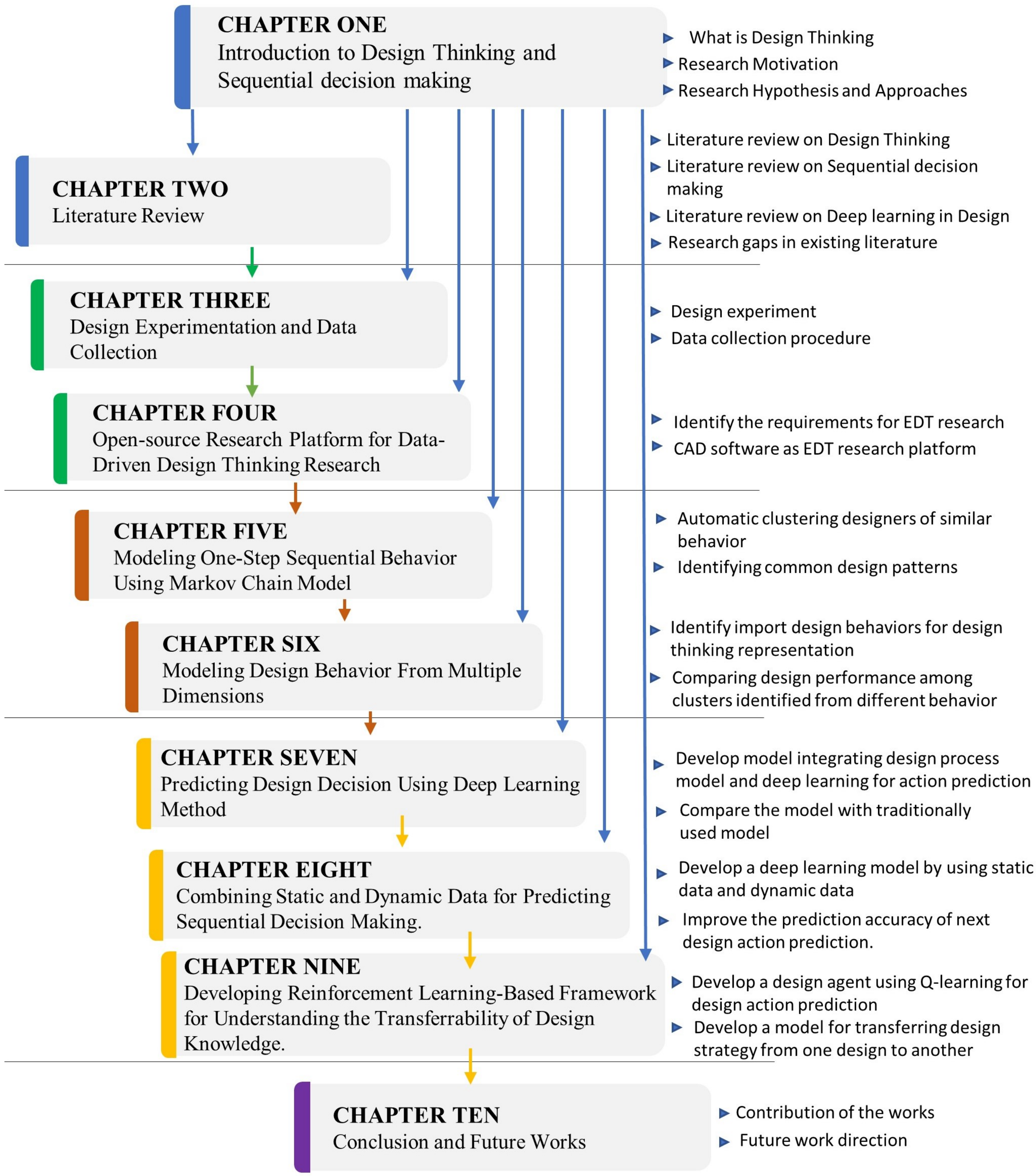


Figure 1.2: Thesis road map

2 Literature Review

2.1 Research on Engineering Design Thinking

Extensive studies have been conducted to study design thinking. These studies adopted various ways to represent design thinking, such as by using cognitive study (e.g., protocol study, controlled experiment), physiological measurement (e.g., eye tracking, heart rate, electrocardiography (ECG)), neurological signals (e.g., electroencephalogram (EEG), functional magnetic resonance imaging (fMRI)) [13]. Protocol analysis relies on observation and can be categorized into in-vivo studies (e.g., the think-aloud method [14]) and in-vitro studies (e.g., interviews with designer [15]). The observational data needs to be transcribed, segmented and coded, and then post-analyses can be performed to generate insights into design thinking. Typical topics of study include design creativity [16, 17, 18, 19] and fixation [20, 21, 22], example modality [23, 24, 25], the role of sketches [26, 27, 28, 29], and differences of thinking patterns between experts and novices [30, 31, 32]. Since the coding scheme is a critical step, extensive research is carried out in evaluating different coding methods. Design data are encoded by these ontological design model (e.g., function-behavior-design (FBS) design process model), which are collected from protocol study or controlled experiment [33]. These design data are typically designers performed actions [34] and are further encoded to a deeper understanding of design thinking [35]. The encoded design data is analyzed by different computational methods in order to represent design thinking. For example, the first-order Markov chain model representing one-step sequential decision-making behavior is utilized to study design patterns [36, 37]; the hidden Markov model is used to identify hidden design states [34]; In some studies, sketch data are collected besides the verbal and design action data [38]. Sketching is further encoded using different sketch coding methods (e.g., C-sketch method [39]) to represent design thinking. Controlled lab experiments are very effective for validating the causality between factors of interest and design outcomes. As experimental settings need to be well designed beforehand, controlled experiments often offer greater intrinsic validity [40]. The results are often generalizable and extensible [41]. Typical subjects of study in this area include the effects of design cost [42, 43] and designers expertise [44, 45, 46, 47] on design outcomes, team effects in design [16, 48, 49], analogical reasoning [50, 51], and provocative stimuli [52, 53]. The design field recently has a trend of using gamified design scenarios [43, 54, 55, 56, 57] to study design behaviors and thinking.

Design thinking is also studied using various physiological measures such as eye-tracking, ECG, and facial recognition. In the eye-tracking method, eye-tracking devices and software capture designers eye movement and provide gaze points and heat maps of areas of interest [58]. Both the heat maps and gaze points, thereby, represent designers thinking. This method mainly analyzes how much attention designers put on the area of a specific design object, and the data have been used to study design creativity [59] and how designers analyze the functionality of a design object [40]. Using ECG, heart rate variability (HRV) signals can be recorded and connected to mental stress [60]. HRV is measured during the different design segments, and the corresponding mental stress is measured. Different designers show different patterns of stress according to their design thinking.

Data collected from neurological studies try to connect design thinking and brain activity. The two most popular methods for neurological studies are EEG and fMRI. While EEG measures neural activity via the identification of electrical current, fMRI measures brain activity by the brain's blood flow using a magnetic field [13]. From the EEG data, the power spectral density of brain waves is also measured, and the correlation between design activity and brain waves is analyzed [61]. Data from fMRI are images of the brain at cross-sections that provide visual reasoning, such as brain activation patterns during design ideation [62]. Recent studies conducted by neurocognition scientists indicated that when designers engaged in divergent thinking, different cognitive domains were activated with the tasks that require analysis during the engineering concept generation [63]. Design neurocognition researchers also have successfully encapsulated the cognitive functioning behind engineering design [64]. This empirical research confirmed that design thinking is not merely an abstract construct. However, the external design behavior regulated by different cognitive processes involved during the search of design solutions requires further investigation through the study of design actions [65].

2.2 Design Thinking Studies Using CAD Software and Non-Intrusive Data Logging

Non-intrusive data logging is a method that automatically logs a designers actions in real time as they use an interactive simulation or design environment without interrupting their design process. On one hand, this can be realized by commercial CAD software which often has an explicit data schema for logging designers actions and relevant metadata, e.g., timestamps, and structures of design artefacts, e.g., geometry hierarchy, in generic data storage formats such as XML. Data captured through these platforms can be processed and sometimes analyzed computationally to accelerate research. For example, Jin and Ishino [66] proposed a data-mining framework, DAKA, which can extract designers design activity and knowledge from CAD event data. Gopsill et al. [67] used CAD as a sensor to collect design action logs and studied micro design patterns which showed the implications of operations, such as deletion and reversing, in design iteration. Sen et al. [68] presented a non-intrusive protocol study with the aid of a software on measuring information content when designers perform free-hand sketching of design concepts. On the other hand, many researchers have developed their own applets for data collection. For example, in order to explore the design heuristics and sequential design patterns, McComb et al. [69] collect design behavioral data with two configuration design experiments with the aid of self-developed applets for truss design and cooling systems design. Sha et al. [43] developed an economic decision game applet based on z-Tree and studied the effects of design cost on designers sequential decision-making under competition.

Table 2.1: Representative research platforms for EDT studies based on the literature review

Reference	**Design problem**	**Design platform/ software**	**Data format**	**System or component design**	**Design information data**	**Built-in experiment materials**
McComb, et al. 2015 [49]	Truss design	Self-developed and closed source	Not reported	System - configuration design	Not reported	External tutorial; No built-in materials
Yu, et al. 2016 [70]	Desalination system/sea water reverse osmosis	Self-developed and closed source	Not reported	System - parametric design	Design parameter values. timestamp	External tutorial; No built-in materials

Table 2.1 (Cont.)

Reference	Design problem	Design platform/ software	Data format	System or component design	Design information data	Built-in experiment materials
Egan, et al. 2015 [71]	Myosin design	Self-developed and closed source	Not reported	System parametric design	Text. Design parameter values	External tutorial; No built-in materials
McComb, et al. 2017 [69]	Truss design and home cooling system	Self-developed and closed source	Not reported	System configuration design	Design actions	No built-in materials
Jin, et al. 2006 [66]	Car front door	Commercial and open source	Not reported	Component design	CAD drawing commands	No built-in materials
Gopsill, et al. 2016 [72]	Pulley	Commercial and open source	Not reported	Component design	CAD drawing commands	No built-in materials
Ritchie, et al. 2008 [73]	Cable organization system	Commercial and closed source	Yes (XML)	Component design	Design actions in virtual reality	No built-in materials
Sha, et al. 2015 [74]	Function minimizes	Self-developed and closed source	Not reported	Component or System - parametric design	Design parameter values	No built-in materials
Sivanathan, et al. 2015 [75]	Bracket support	Commercial and open source	Yes (XML)	Component design	Design actions and video	No built-in materials
Sen, et al. 2017 [68]	Burger maker	Commercial and closed source	Yes (Excel)	System conceptual design	Design sketches. timestamp	No built-in materials

Table 2.1 (Cont.)

Reference	Design problem	Design platform/ software	Data format	System or component design	Design information data	Built-in experiment materials
Toh, et al. 2014 [23]	Milk froth device	No software platform	Not reported	System conceptual design	Sketches	No built-in materials
Gero, et al. 2018 [17]	Wheelchair assist device	No software platform	Not reported	System conceptual design	Sketches and video	No built-in materials
My Dissertation (Energy3D)	Solar energy systems	CAD-based platform	Yes (JSON)	System multiple design stages	Design actions, design config, text, artifact, etc.	Built-in experiment materials

Additionally, logged data may be processed in real time to offer designers immediate feedback or suggestions for advancing their design process [76, 73]. Sivanathan et al. [75] extend the data logging method to what they call ubiquitous multimodal capture that incorporates CAD logging, keyboard and mouse logging, eye tracking, screen and environment video, galvanic skin resistance, electroencephalogram, and electrocardiogram. They demonstrate the feasibility of this collection scheme through case examples of bracket design and collaborative design review. While ubiquitous multimodal capture collects extensive data from a designer, it comes at the cost of being disruptive for the designer and expensive to implement. In Table 2.1, we summarize the representative research platforms used in design literature.

2.3 Studies on Sequential Decision-Making and Design Behavior

Several studies have been done in the engineering design field to explore the sequential patterns, optimize the sequence of design task, and finding heuristics from sequence learning. Particularly, a large number of studies have been conducted based on the Markov chain model for sequence learning. For example, in order to compare designers sequential design behaviors in three different domains, including architecture, software design, and mechanical design, the function-behavior-structure ontology and the first-order Markov chain [33] were adopted. The designers are asked to solve their own domain-specific design problems. The second-order Markov chain model was also used to explore the effect of previous experience and design knowledge on design sequence

[77]. In order to study designers sequential learning strategies, McComb et al. [78] used a Markov chain model in a truss design problem. Their results indicate that the first-order Markov chain better represents designers action sequences. In a later study, they used the hidden Markov model (HMM) to study the patterns of sequential design state in the same design problem. They found four hidden states in the configuration design and observed that designers used the first two states to topology operation, third state to spatial, and the fourth state to parameter operation. The trained HMM model was then utilized to compare the design processes of the high-performing group and low-performing group [79].

To computationally model designers sequential search process, there have been studies based on Bayesian Optimization (BO) framework. For example, to mimic the human searching strategy, Sexton and Ren [80] developed a searching process using the BO algorithm, which can replace human solvers from a design process. Sha et al. [43] also integrated the Weiner process BO with game theory to study designers sequential decisions in a one-one-one competition for monetary reward. Some studies have also used the Gaussian process-based model [81] and descriptive models based on expected utility (EU) maximization [82] to understand human design strategies in the sequential information acquisition decision making (SIADM) scenario. In order to quantify the impact of designers domain knowledge and problem framing, Shergadwala et al. [83] developed a SIADM framework incorporating expected improvement (EI) maximization and optimal one-step look-ahead strategy. The framework is applied to a motor track design problem and found that problem framing impacts designers knowledge as well as their performance. Later, the framework is extended as a Strategic-SIADM model to understand the influence of competitors past performance on individuals design behavior and outcomes [84].

Prior studies on sequential design processes have also been focused on project task level in support of product development and project management. For example, the design structure matrix (DSM) [85] has been used to study task sequencing for identifying the sequence that minimizes expected project completion time. Some other work has been grounded in theoretical processes. For example, Miller et al. [86] use multi-objective formulations to study the design process sequentially advancing through to smaller sets of alternatives using models of increasing fidelity. In addition, optimization approaches, such as the expected value of perfect information [87], genetic algorithm [88], and optimal learning [89], have been utilized in studying optimal design sequences. However, these studies are fundamentally different from the presented work in that they formulate a design problem and cast it into a sequential decision process to be optimized with normative models. In this dissertation, however, we focus on the sequential decision-making of human designers. It is about the actual actions that designers sequentially take. By modeling and analyzing such a design sequence at a fine-grained resolution, it is expected that insights and new knowledge regarding the designers thought process can be obtained.

2.4 Deep Learning Methods in Design Research

In recent years, deep learning techniques have shown their promise in the design field to solve different design problems, including design optimization, design ideation, and design behavioral modeling. For example, Raina et al. [90] developed a two-step deep learning framework. A convolutional neural network-based auto-encoder is used in the framework to map the images of design to a low dimensional embedding to generate design without specific design operation (i.e., adding any particular design component). In the second step, the derived embedding and a rule-based image processing inference algorithm are used to output the operation, construct the structure, and iteratively improve the design. The resulting design is found to have a better factor of safety and strength-to-weight ratio over human designs. Oh et al. [91] developed a framework where topology optimization and boundary equilibrium generative adversarial network (BEGAN) are used iteratively to generate new designs. The proposed method is applied to a case study on a 2D car wheel design. Stump et al. [92] developed a method for optimizing the structure and attributes of sailboat design. By embedding spatial grammar in character recurrent neural network (char-RNN), the structure of the sailboat is optimized in a physics-based game engine, Unity3D. In a similar case study, reinforcement learning (RL) is used to minimize the time of the sailboats travel path. Table 2.2 shows the summary of these relevant studies.

Table 2.2: The summary of the relevant literature on deep learning in engineering design

References	Data type	Used deep learning model	Research objective	Design context	Number of available design operations
Raina et al. 2019 [90]	Images of sequence of design artifacts	Convolutional neural network	Generative design and optimization	Truss design	9 designs operations
Oh et al. 2019 [91]	Images of sequence of design artifacts	Generative adversarial network	Topology optimization	Car wheel design	N/A

Table 2.2 (Cont.)

References	**Data type**	**Used deep learning model**	**Research objective**	**Design context**	**Number of available design operations**
Stump et al. 2019 [92]	3D CAD geometry	Char-recurrent neural network	Structure and attribute optimization	Sailboat	N/A
McComb et al. 2017 [69]	Text data sequence of design actions	Hidden Markov model	Identify beneficial design heuristics	Truss design & cooling system design	9 operations for truss design & 7 for cooling system design

Among different deep learning algorithms, Recurrent neural networks and its variants are particularly effective to handle sequential or dynamic data. For example, Almeida & Azkune [96] propose a deep learning architecture based on long short-term memory (LSTM), a variant of RNN, to predict users future activities during a day, such as sleeping, walking, and eating, with sensor data collected from wearable devices. The model can successfully predict a users next action and help identify anomalous behaviors. While predicting future events, the majority of the RNN-based

Table 2.3: Comparison of the studies that combine static data and dynamic data

References	**Dynamic data**	**Static data**	**Text vectorization and dimension reduction**	**Comparative study included**
Esteban et al. 2006 [93]	Clinical log event	Patients demographics	Word embedding	Yes
Markis et al 2017 [94]	Drum sequence	Bass information	No dimension reduction	No
Sharma 2015 [95]	Drawing order of image	Pixel location	No dimension reduction	No

models merely utilize dynamic data. There are only a few studies that use both static and dynamic data for prediction by leveraging a separate algorithm, such as the feed- forward neural network (FNN) or random forest model. For example, Esteban et al. [93] presents a deep-learning approach that takes static information (i.e., gender, blood group) into an FNN and dynamic information (patients visits at different time) into an RNN to predict future clinical events. The model is applied to the data collected from the kidney failure patients and predict among three possible endpoints that would occur after kidney transplantation.

In a similar study [94], but in musical research, a deep-learning architecture is developed by combining LSTM and FNN to generate drum sequences. In this architecture, drum sequences, i.e., the dynamic data, collected from three bands, are fed into an LSTM layer while FNN takes the bass information as static data. The outputs of both layers are then merged to produce the final sequence. To automatically recognize hand-written digits, Sharma [95] extracted static data and dynamic data from the digit images. The static data includes white in a square, i.e., a horizontally, vertically, and diagonally divided region in the pixel. The dynamic features are extracted from the drawing order of the corresponding image of the digit. The support vector machine is used to classify the digit with these concatenated set of features for the recognition of hand-written digits. Table 2.3 shows the summary of the studies that use both static and dynamic data in the deep-learning methods. But none of them is related to engineering design.

2.5 Transfer learning in RL and design

Transfer learning, which emerged as a research area in the machine learning domain, is a method where the goal is to obtain pre-trained values (knowledge) in a computational model and to use them for a new problem [97]. Transfer learning saves training time and resources. Typically, during model training, hyperparameter tuning and model training takes a lot of time. Transfer learning reduces time in new problem solving by reusing an already trained model from a different problem instead of training a new model and choosing hyperparameters from scratch.

Like other machine learning areas, Transfer learning is also ubiquitous using RL. Transfer learning in RL can be categorized mainly into three major groups that include parameter transfer, instance transfer and representation transfer [97]. In parameter transfer, the target task can use the RL parameters (i.e.,initial values or learning rate) according to the source tasks. Parameter transfer is suitable when the source and target tasks share a common state action space [98]. [99] introduces a Variable Reward Hierarchical Reinforcement Learning (VRHRL), a parameter transfer method, which uses previously learned policies to speed up and improve the result. They assume that the reward function is a linear combination of reward weights across Markov Decision Process

(MDPs). In another study, Attend, Adapt, and Transfer (A2T), a deep RL model is introduced that is able to select and transfer from multiple source tasks for different parts of the state space of the target task in the same domain. This model transfers the policies or the value function to the target MDPs. Their results indicated that the model successfully transferred knoweldege selectively from multiple source tasks [100]. In the instance transfer, samples from different source tasks are used to learn the target task. For example, [101] transferred trajectory samples from the source task and used them in the model of new tasks to simplify the estimation of the model.

In the representation transfer, the RL agent learns a representation of the source task and performs an abstraction process to fit it to the target task. In this process, studies have used neural networks for feature abstraction [102, 103], while others strategies are also explored. For example, a reward shaping approach is used for knowledge transfer [104].

Although transfer learning has been used in many applications, it has rarely been studied in design behavior research. Raina et al.[90] propose an approach to transferring design strategies between similar design problems in the same context. In their study, a hidden Markov model (HMM) is used to learn the design strategy. They use the CISAT framework as an agent to transfer the learned design strategy from a home cooling system design problem to a scaled down version of it. The results indicate that the agent performs better in different problems, especially at the beginning stages of the design process rather than the later stages.

2.6 Research Gaps

Although researchers adopt many techniques for data collection and study design thinking and decision making, there are some fundamental gaps in the existing literature. The gaps are described below:

- Although CAD-based non-intrusive data collection methods have proved to be efficient, there are lacking that hinder its full potential in design thinking and decision-making studies. For example, many self-develop applets have limited functionalities and only address a particular design phase, e.g., conceptual design. In addition, commercial CAD software is a tool and not designed for research purpose per se. The data collected from such software are typically drawing or sketch commands (e.g., circle, extruding), and are not able to produce a continuous flow of research data in a complete design process. Yet, design has a life cycle and contains many design stages, such as concept generation, preliminary design, embodiment design, engineering analysis, design validation, etc. Many facets of EDT due to the systems design essences (often require systems thinking) can be hardly assessed. Therefore, the insights obtained from the research based on those platforms are difficult to draw generic insights. The data collected from self-developed applet is also ill-structured without using standard data structure and schema. This creates burdens for data processing and sometimes causes missing data, which inevitably hinders the major effort at the core of research. More importantly, these applets are often in closed form and not accessible from public, which causes barriers in repeating and reproducing the research findings for cross-validation.

- Although deep learning methods serve as the core of the research approaches in several studies in the design field, the research objectives are fundamentally different from this dissertation. Current deep learning-based methods aim at improving or optimizing a design output or generating new designs by training a neural network that learns from existing design artifact data. For example, Raina et al. [90] uses 2D images of the designs as their input and uses the CNN model to learn truss design structures that can yield high design performance. In contrast, the current study aims at understanding designers thinking in an engineering system design process and learning the beneficial design sequences for an in-depth understanding of their thought processes. Because of this motivation, the integration of a higher level of abstraction of the design process (often known as design ontology) is needed to transform the design action space to the design process stages (design thinking space) so that a sequence learning can be applied to learn the beneficial yet hidden patterns of designers thinking.

- Even if the existing studies use sequential actions to computationally model and study designers sequential decisions, little research has been studied incorporating design-related attributes (static data). There is a large number of different types of design actions engineering design context. On one hand, this causes high-dimensional text data which challenges the modeling and deep learning. On the other hand, each action is chosen by the designers thus reflect their thinking/objectives the moment when they made such a choice. This feature requires the knowledge of appropriate design process model to abstract the design action to the level of the design process in order to better study human design thinking and behaviors. The complexity associated with the sequential data in engineering design is more than that of the data handled in the existing literature. For example, in (Esteban et al. [93]), each data in the sequence represents one of the three endpoints that a kidney patient may face after kidney transplantation, i.e., kidney rejection, kidney loss and death of the patient. The inputs are laboratory analysis results obtained at different dates. Such sequential data are objective values and do not indicate any human thinking, thus not require a meta-model of design thinking for characterization. In engineering design, while design actions can be automatically logged from tools that designers interact with, their demographic information and attributes are often not collected or very limited. This is different from the existing literature, for example, in the application of clinical events, patients information can be readily obtained because their personal information needs to be provided at the time of visit. Little study was conducted in the design field that provides a solution to combine both types of data for the prediction of sequential design decisions. There is a knowledge gap that must be filled in order to better facilitate design research where much static information of human factors cannot be neglected.

- Though the study of reflective thinking is a growing trend, very few studies have been conducted on design reflection [105]. Goldstein et al. [105] use designers electronic notepad and pre-test and post-test to study designer reflective thinking and found that moderately reflective students understood design activities better than those with high or low reflectivity. Even though many studies on design behaviors have been conducted, most of them focus on a particular design behavior at a time. However, design thinking is not merely a particular design behavior; rather, it is an abstraction of design behaviors from multiple dimensions. Therefore, to a deeper understanding of design thinking, a study on different design behaviors is needed.

- Although transfer learning is leveraged in many areas to solve data scarcity, time and resource issue, the application of transfer learning to design problems is an understudied area where benefits and limitations are unknown. In particular, it is unknown whether the design knowledge learned from a machine-leaning model is transferable or not. Research that has been conducted on transferring design knowledge (i.e., Raina et al. [90]) is fundamentally different from this study. There are three main differences between this study and the work presented by Raina et al.[90]). First, they uses the hidden Markov model, while in this dissertation, we use RL to train the design agent. RL reinforces frequent design patterns and uses an optimal policy to improve prediction accuracy, while HMM predicts the sequence without the reinforcement mechanism. Second, in this study, the target task is significantly different from the source task. However, they use the same problem as the source problem with a scaled-down version as their target task. Third, we adopt a higher-level design process model to represent design knowledge, where Raina et al. [90] focus on individual actions that are limited to be generalized beyond the specific design problem.

3 Overview of Experimentation and Data Collection

To answer the two RQs and test the corresponding hypothesis, human-subject experiments were conducted under different design contexts from which the design-related data, such as design actions, CAD models of design artifact, design performance scores, etc., were collected. Particularly, in answering RQ1, we conducted two design experiments. The first one is a controlled design experiment, and the design problem used in that experiment is a solarized home design problem. The second one is a field design experiment where a design challenge is created for participants to design a solar energy systems solution for an apartment complex on their campus. In the controlled experiment, the design is done in laboratory settings or classroom settings, and participants do their designs under the investigation of researchers [10]. A standard procedure is used where there is precise control over extraneous and independent variables. In answering RQ1, the controlled experiment is used to identify useful design patterns. Since in the controlled experiment, there are constraints on the variables (i.e., independent and extraneous), designers only focus on the given design variables and exploit the design space with those variables. Therefore, it supports us in identifying useful design patterns and design heuristics. However, a field experiment is done in a real-world or natural setting where there are limited restrictions on design variables. Therefore, more authentic and detailed behavioral data can be generated during the field experiment. We adopt a field experiment in support of the data collection so that we can implement the proposed approaches to the model design behaviors from multiple dimensions. Since designers can more freely explore the design space in a natural setting than in a controlled setting and spend more time solving the design problem, the logged design action data could better reflect designers behaviors.

In answering RQ2, in addition to the solarized home design problem, we developed another controlled experiment with a different design problem, solarized parking lot design. In the solarized home design problem, designers are asked to design a solarized home with a budget of $200,000. In the solarized parking lot design, the challenge is to build a solarized parking lot with a budget of $1.5M. The objective of both challenges is to maximize the annual net energy (ANE) with the given budget. We have set specific design requirements and constraints to the participants so that the participants can start the design with a focus on configuration and parametric design. We limit the number of design variables and design constraints so that the experiments remain in a controlled manner. Though both are conducted in controlled settings, they are fundamentally different in terms of design complexity. For example, the solarize home design is more complex in the sense that it has more design variables and more complex couplings between variables than that of the parking lot design problem. Therefore, they are useful in testing the generality of the proposed approaches and methods for the RQ2. Also, we would like to investigate how the prediction accuracy would be different under different scenarios with different design complexities. Table 3.1 shows the overview of the design experiments, their goals, the related tasks, and the expected outcomes.

Table 3.1: Design experiments, tasks associated with the experiments and the expected outcome

Design context	**Goal**	**Tasks**	**Expected Outcomes**
Solarized home design (**RQ1**)	To collect design data from a controlled experiment and implement the proposed approach for identifying designers sequential behavior and beneficial design patterns and heuristics	1. Develop a complex system design problem where designers focus on building a solarized house, optimize its cost, and maximize its energy output. 2. Setup the design environment in laptops, advertising the design challenge, and recruiting participants. 3. Formulate design requirements, design constraints, and relevant tutorial sheets.	1. Collection of design behavior data in the forms of action sequence, design documents, design artifacts. 2. Validation of the proposed approach to identifying sequential design behaviors, commonly used design heuristics, and patterns.

Table 3.1 (Cont.)

Design context	**Goal**	**Tasks**	**Expected Outcomes**
Solarize UARK campus (**RQ1**)	To collect design data from field experiment in support of the implementation of the proposed approaches to model design behavior from multiple dimensions and research their relations with designer performance.	1. Develop a design problem that is in real-world settings and provide more authentic design data regarding design thinking and sequential decision-making. 2. Develop the complex system design in such a way that it focuses mainly on convergent thinking (i.e., optimizing cost, energy output, and payback period).Implement psychological experiments to capture designers psychological attributes. 3. Implement psychological experiments to capture designers psychological attributes.	1. Collection of the dataset which includes both design-related data (i.e., actions sequence, design documents, and design artifacts) and psychological test data. 2. Validation of the proposed approaches in characterizing and modeling the design behaviors from multiple aspects, such as design action frequency, the context of design actions, sequential design decisions and actions. 3. Validation of the proposed approach that predicts designer performance based on the design behavioral features.

Table 3.1 (Cont.)

Design context	**Goal**	**Tasks**	**Expected Outcomes**
Solarized home design & Solarized parking lot design (**RQ2**)	To collect design data from controlled experiments and implement the proposed approach to predict future sequential design decisions.	1. Develop two design problems with different design complexity in terms of the number of design variables and couplings between those variables. 2. Formulate the corresponding design requirements, design constraints, and design documents such as tutorial sheet, scientific knowledge card, etc. 3. Design a record sheet for the designers to input their design notes and design performance data (i.e., design objective values).	1. Collection of datasets which includes design action sequences, design artifacts in form of CAD models, and design performance data (i.e., design objective values). 2. A set of predictive models and validation of the proposed approaches that integrate designers' attributes and design preferences into those models to improve the prediction accuracy of design decisions. 3. A design agent that mimics human design behavior and validation of the transferability of design knowledge.

We follow end to end procedures to conduct the design experiments. Necessary documents such as consent form, design statements, tutorial sheets are prepared for each of the design experiments separately (see appendix for details). For the controlled experiments, interested participants sign up through a link and choose their preferred session. At the beginning, the designers takes their seats with the corresponding laptop number. The participants were indexed based on which session they were in and which laptop they used; For example, A02 indicates a participants where the participant was in Session A and sit in laptop #2. The experiments are conducted in two phases: pre-session and in-session. The pre-session is 30 minutes for participants to practice Energy3D with the built-in design tutorials. In addition, we also provide tutorial sheet for particular design experiments. The pre-session is designed to account for the learning curves of humans. The data generated in pre-session is not used for analysis. At the end of the pre-session, the participants are guided to transition to the in-session stage. The in-session stage lasts about 1.5 hours. The design statement and the design requirements are provided at the beginning of this session, and a record sheet is provided for participants to record the ANE and cost whenever they iterate their designs. Monetary rewards are provided at the end of the session to incentivize the participants to search the design space as much as they can. The participants are rewarded based on the amount of time they spend as well as the quality of their final designs, which are quantified by the ANE value and the construction cost.

Similar to the controlled experiment, in the field design experiment, participants also sign up to enter the design challenge. Participants begin the procedure by receiving an introductory presentation from us. In the presentation, we go through the design problem, design requirements and constraints. Additionally, we also presented essential tips and tricks and a brief tutorials on Energy3D. At the end of the meeting, participants are provided a flash drive containing the presented documents and the design problem in the Energy3D file format. The participants are provided seven full days to complete the task on their own time. Once participants completed the design challenge, they submit their solution with the flash drive. After submissions are rated by the research team and the winners are decided, the challenge end with an award ceremony to announce the winners.

4 Open-Source Research Platform for Data-Driven Design Thinking Research

In this chapter, we introduce our research platform for EDT studies based on Energy3D and discuss the key features that make it a suitable and powerful tool in supporting ETD research. Before demonstrating the details, we first summarize our view on the data requirements that shall be met for EDT research in order to address the limitations of existing methods identified in Section 2.6.

4.1 Data Requirements for EDT Research

To successfully execute the proposed research and achieve the objective, the data is critical and specific requirements deserves careful attention. We identified five requirements based on our literature review performed in Chapter 2.

1. Intra-stage and inter-stage design iteration. Design iteration does not only occur within each stage but also between stages [106]. For example, designers often utilize science simulation to refine their designs in concepts generation [107]. The decisions made during such an iteration play a vital role in assuring a successful design. A tool that supports the collection of design process data and design actions in both intra-stage and inter-stage iterations is needed.

2. High fidelity. In a design process, ad-hoc decisions are often made. Unnoticeable actions could be nontrivial information reflecting useful decision-making strategies. It would be ideal if every single movement of designers can be recorded. The data should be a collective memory of the complete output and all iterations in design.

3. Non-intrusive. Intrusive data collection (e.g., interviews) is time-consuming, and thus restricts research scale [55, 108]. Such a process could easily add cognitive load to designers, thus possibly contributing biases toward the observed behaviors [109]. These limitations can diminish the validity of the data.

4. Rational behavior. Most decision theories assume rational behaviors, but designers have bounded rationality [110]. When collecting design behavioral data, designers irrationality should be accounted for and decision-supporting tools (e.g., simulations) shall be leveraged to inform rational decisions to improve the quality of design data.

5. Multiple forms. The data should be a combination of operational, textual and even video data to support the cross-validation of research approaches or methodologies.

4.2 Using Energy3D as a Research Platform for EDT Studies

With the aim of meeting these requirements, we introduce a new research platform based on Energy3D, a computer-aided design software developed by Xie [111], one of our team members. It was developed as a tool for solar energy systems design and analysis as well as for K-12 education research. To make it applicable to support engineering design research, specific features, such as built-in experimentation, tutorial and templates, new computational modules, and additional data collection methods, have been added. In summary, as a platform for design research, Energy3D has unique features in three aspects:

1. **Feature for data collection**

   First, Energy3D can continuously and automatically log and sort every user action and design snapshots (computer models, not images) in a fine-grained resolution. These data represent

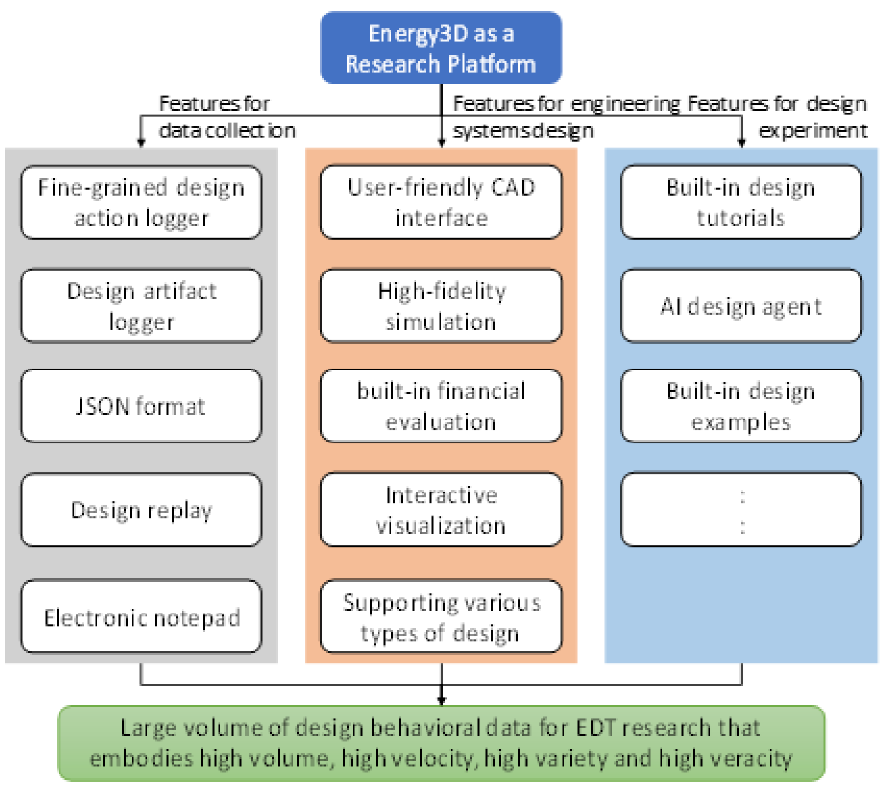


Figure 4.1: Unique features of Energy3D in supporting engineering design thinking research

the smallest transformation possible on a design object that changes how it looks or performs. That means the design process and even the design artifact can be entirely reconstructed without losing any important details. Therefore, it works as a sensor of design behaviors so as to capture the design processes in detail, and in a non-intrusive manner, i.e., designers are not aware of the data collection and can concentrate on their design activities without any hindrance to their design thinking.

Second, Energy3D logs the data in JavaScript Object Notation (JSON) format. JSON is a generic machine-readable data storage format that relies on two common data-structures, arrays and key-value pairs, to encode a variety of data schemas. In Energy3D, each logged action contains the action itself, e.g. adding a wall, and metadata about the time and date when the action was taken. Central design attributes associated with each action are recorded as well, such as the size of a window or the results of an energy simulation. Such a standard data format makes it possible to translate any design activity, logic, or strategy into computer code and vice versa. JSON data can be readily processed by most programming languages and statistical platforms like R, making it convenient for researchers to analyze the data with their preferred tool set. Standardization and automation make the design research cost-effective and scalable.

Third, Energy3D stores a rich blend of both qualitative and quantitative data. In the JSON file, textual design actions are stored as qualitative data and values of design parameters are stored as quantitative data. For qualitative data, Energy3D has an electronic notepad.

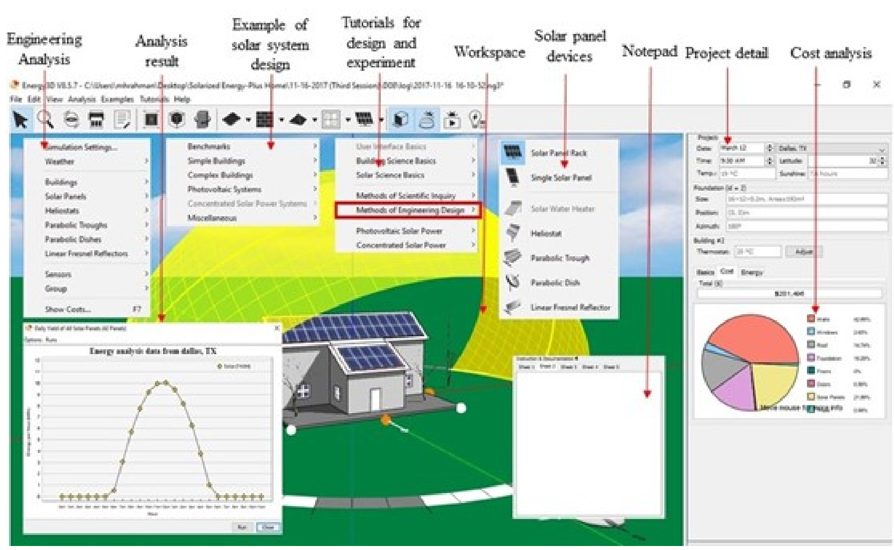


Figure 4.2: Unique features of Energy3D in supporting engineering design thinking research

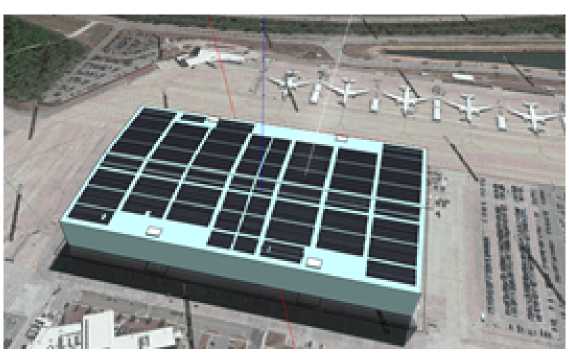
During a design, designers can write down important ideas, findings, and thought processes. The collection of these data is particularly beneficial to research on systems design thinking where both quantitative and qualitative skills are required [74].

Fourth, in addition to CAD modeling, Energy3D has built-in modules of engineering analysis, scientific simulation, and financial evaluation that realize a seamless design environment. This ensures the collection of design data during intra-stage and inter-stage iterations, for example, how designers make decisions with economic considerations, i.e., design with rationality. These data are important to the study on designers thinking in complex systems design and the role of system thinking in engineering design.

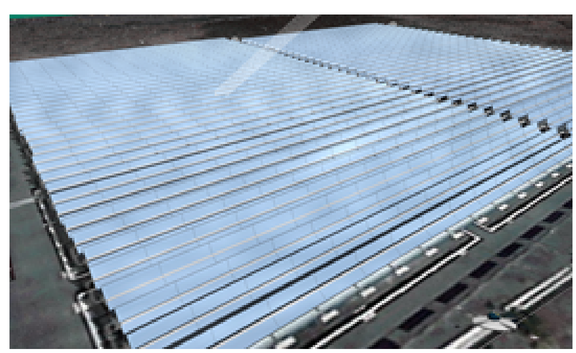
2. **Features for engineering systems design**

   The interface of Energy3D is intuitive to operate [112]. Energy3D encompasses several pre-designed components (i.e., doors, window, solar panel, etc.) that ease the design difficulty such as drawing components from scratch. This ensures participants can focus on the design process and employing design thinking instead of time-consuming drawing process. In addition, easy operation will help shorten participants learning curve, which often influences the validity of behavioral data in design research.

   It is worth noting that, simulations in Energy3D are very accurate and trustworthy as it provides real-world design configuration and materials [112]. For example, the building simulation engine is calibrated with DOE's BESTEST benchmarks. This feature ensures authentic and high-fidelity design practice. Moreover, during simulations, Energy3D graphically illustrates the results through interactive visualization and animation, which allows designers to easily and efficiently get formative and immediate feedback. This is critical to their rational decision-making and exploration of design space in engineering systems design.



(a) A rooftop photovoltaic system for Boeing's South Carolina factory



(b) Parabolic troughs in Hawaii

Figure 4.3: Different designs supported by Energy3D

Energy3D supports various solar energy system designs (see Figure 4.3). Design can be conducted in various contexts with different options of solar harvesting devices including solar panel, heliostat, and parabolic trough, and different solar panel brands. With these capabilities, researchers can create different experiments covering a wide range of design scenarios in different levels of design complexity, such as single component design, geometric design, layout design, material design, and architectural design.

3. **Features for human-subject experiments in design research**

   To facilitate experiments for design research, Energy3D contains many built-in tutorials for designers to get acquainted with the domain knowledge of solar science, building science, and engineering design. These tutorials can be used as pre-session before experiment in order to account for the variation of learning curves among participants. In addition to these tutorials, a set of design experiments have also been made publicly available through the authors websites [77, 78]. The researchers can easily adapt these examples to create new ones for their own research purpose and data collection.

With all the above features in three aspects, as summarized in Figure 4.1, Energy3D can support the collection of a large volume of fine-grained design behavioral data which is essential to big data mining and machine learning of EDT. Such fine-grained data possess all four characteristics of big data [113]: 1) High volume. A large amount of design process data

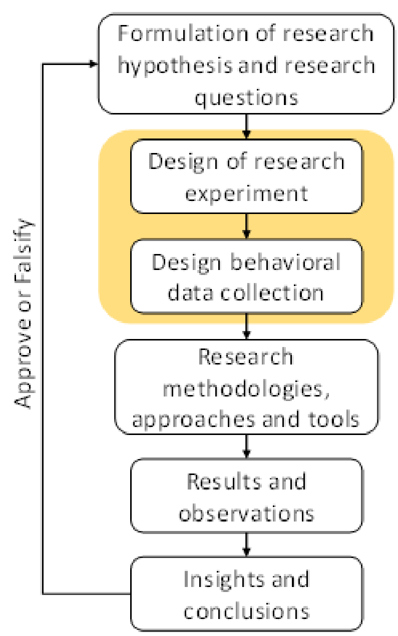


Figure 4.4: A typical cycle for data-driven engineering design thinking research

will be generated. 2) High velocity. One of the characteristics of big data is velocity which means how fast the data is collected. Energy3D collects, processes and visualizes data in real time at the scale of seconds. Such an improvement on the continuity of behavioral data can help improve the understanding of the flow of design thinking. 3) High variety. The data encompasses multiple forms involving design actions, parameters, analyses and simulations. 4) High veracity. The data is comprehensive to ensure fair and trustworthy assessments of designer performance. These big data have the potential to yield direct, measurable evidence of design thinking at a statistically significant level. This is fundamentally different from existing studies [66, 72, 76, 73] using CAD logs that contain merely drawing commands. Xie and colleagues prior work [114, 115, 116] has shown that the data collected from Energy3D is capable of measuring the level of engagement, revealing gender differences, and distinguishing the iterative and non-iterative cycles in design. With Energy3D as the platform, we follow a typical scientific research cycle as shown in Figure 4.4 to conduct EDT research. As indicated in the figure, the step of research experiment and data collection are critical links in this cycle yet their rigor and validity have received little attention. As introduced above, those unique features of Energy3D will provide researchers with strong support.

4.3 Conclusion

With the growing trend of leveraging data analytics and machine learning approaches in engineering design research, there is a need to create a research platform that enables the sharing of benchmark problems and testbeds, and ensures the quality of datasets for valid, repeatable and reproducible research. In this chapter, we propose five important data requirements for design-driven EDT studies that must be satisfied in the first place in order to support the scientific rigor. Towards fulfilling these requirements, the authors take a few modest steps in this direction by creating and distributing a research platform using Energy3D a computer-aided energy systems design software. We demonstrate the key features of Energy3D in three aspects: 1) features for data collection, 2) features for engineering systems design, and 3) features for human-subject experiments. The blending of these features can effectively help researchers obtain datasets that satisfy those critical data requirements and exhibit the 4V features of big data, thus make Energy3D a competitive candidate platform for data-driven EDT research. Based on this platform, we collect design data through different design experiments and conduct research that are presented in the following chapters.

5 Modeling One-Step Sequential Behavior Using Markov Chain Model

5.1 Overview

In this chapter, we develop a framework for clustering designers with similar sequential design patterns. We adopt the Function-Behavior-Structure based design process model to characterize designers action sequence logged by computer-aided design (CAD) software as a sequence of design process stages. Such a sequence reflects designers thinking and sequential decision making during the design process. Then, the Markov chain is used to quantify the transitions between design stages from which various clustering methods can be applied. Three different clustering methods are tested, including the K-means clustering, the hierarchical clustering and the network-based clustering. A verification approach based on variation of information is developed to evaluate the effectiveness of each method and to identify the clusters of designers who show strong behavioral similarities. The framework is applied in a solar energy systems design problem energy-plus home design. The case study shows that the proposed framework can successfully cluster designers and identify their sequential decision-making similarities and dissimilarities. Our framework can support the studies on the correlation between potential factors (e.g., designers demographics) and certain design behavioral patterns, as well as the correlation between behavioral patterns and design quality to identify beneficial design heuristics.

5.2 Overall Approach

5.2.1 General approach

In this section, we present our approach to clustering designers sequential decision-making behaviors. As shown in Figure 5.1, we use several data types which are converted from one to another. Each data type is described as below:

- Design action data: In this study, design actions are defined as the design-related operations used in a CAD environment, for example, adding a new component or changing the size of a component.

- Design process data: Design process data is transformed from design action data by a design process model, e.g., the waterfall model or the spiral design model, etc. The design process data has a reduced dimensionality as compared to the design action data depending on the number of processes defined by a design process model.

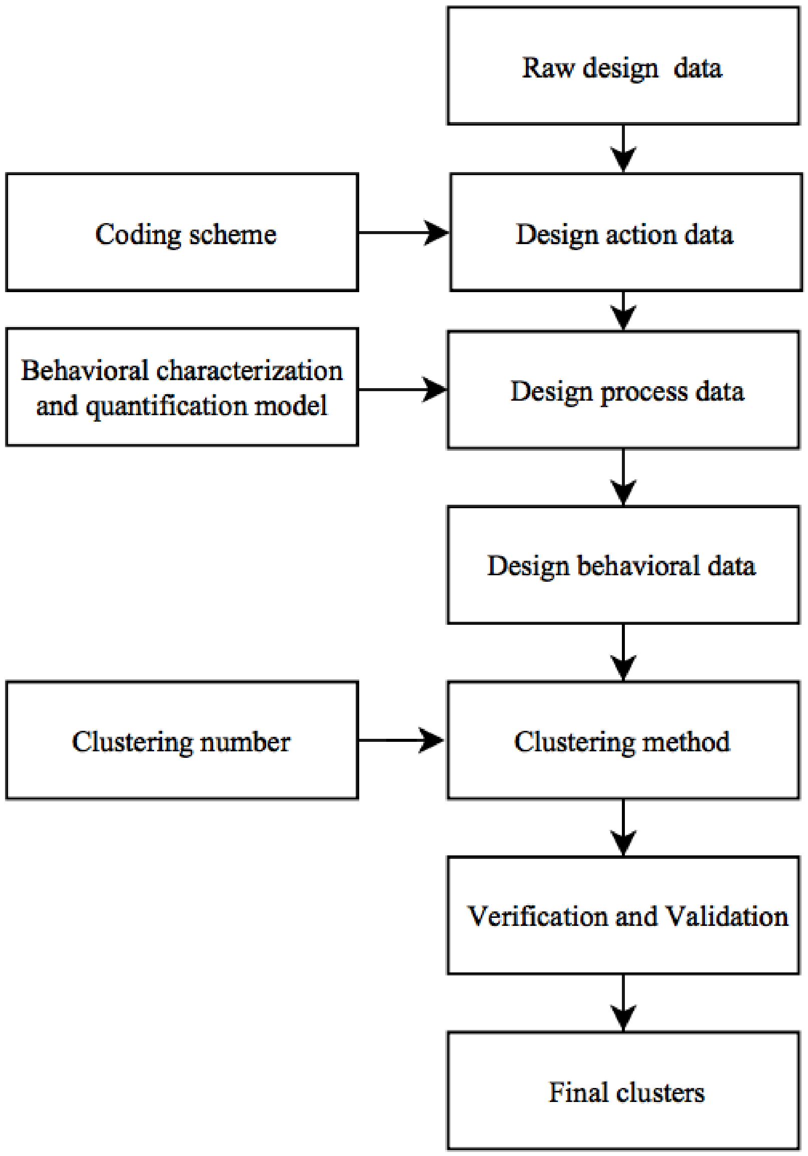


Figure 5.1: The approach to automatically clustering design behaviors

- Design behavioral data: This the data generated from design behavior models by taking design process data as the input. The resulting data characterizes and quantifies design behavioral features. For example, if using Markov chain to study the sequential decision-making behaviors, the design process data will be converted to the transition probability matrix (see Section 5.1 for details), which is regarded as the design behavioral data.

Once the design behavioral data is obtained, different clustering methods can be applied to group designers with similar behavioral patterns. The optimal number of clusters can be determined from a standalone method, e.g., using elbow plot [117] in K-means clustering, or sometimes the clustering methods can automatically determine the best number of clusters. Since different clustering methods usually produce different clustering results, it is important to verify the results from different methods. Therefore, a verification approach is needed to assure the correctness and the quality of outputs. Its worth noting that each of the components in Figure 1 can be programmed and seamlessly connected to turn the approach into an automatic clustering tool. In the following subsections, we present the details for each step.

5.2.2 Characterizing Sequential Decisions Using Markov Chain

In this study, the first-order Markov chain [118] is adopted to characterize a design process transitioning from one stage to the another. A Markov chain is a stochastic process in which a system transitions between a finite numbers of discrete states. The traditional definition of Markov chain is regulated by the Markov property the future state of the process is solely based on its present state. This refers to the first-order Markov chain model [119]. Higher-order Markov models can be developed assuming the next state depends on the current state as well as some number of past states [78]. To define a discrete time Markov chain, we need three components:

- *State space:* a finite set S of possible states of the system.
- *Transition probabilities:* a function $\pi\vdots S \times S \rightarrow R$ such that $0 \leq \pi(a,b) \leq 1$ for all $a, b \epsilon S$ and $\sum_{b \epsilon S} \pi(a,b) = 1$ for every $a \epsilon S$.
- *Initial distribution:* a function $\mu : S \rightarrow R$ such that $0 \leq \mu(a) \leq 1$ for every $a \epsilon S$ and $\sum_{a \epsilon S} \mu(a) = 1$

In order to use Markov chain to study the sequential decision-making in design, some treatments are needed to adapt the concepts of Markov chain. While designers explore a design space, design actions performed at different time spots may correspond to the same design process stage. The sequence of how the design space is explored (design action space) can, therefore, be mapped to a design process (design thinking space), where the Markov chain can be established to model the sequential decision making as a time series of design stages. In such a configuration, the system states in the Markov chain corresponds to design process stages, and the system is, therefore, the sequential design thinking being studied. To support the mapping of design actions to design process stages, a coding scheme (see Section 5.3.3) is developed based on the FBS-based design process model.

Based on the five FBS ontological variables mentioned previously, such a design process model consists of eight process stages: Formulation, Analysis, Evaluation, Synthesis, Documentation and Re formulation 1, 2 and 3. Table 1 defines how these design process stages are derived from FBS ontology. Formulation transforms Requirement (R) into Function (F) and from Function to Expected Behavior (Be). Synthesis generates and tunes Structure based on the Expected Behavior. Analysis is defined as the process which is generated from Structure (S). Evaluation is the comparison between the Expected Behavior and the behavior enabled by the actual structure (Bs). The design process that transitions from Structure is called Re formulation. Depending on which state the process transitions to, three different process stages can be defined. Re formulation 1 is the process transitioning from one structure to a different structure. Re formulation 2 describes

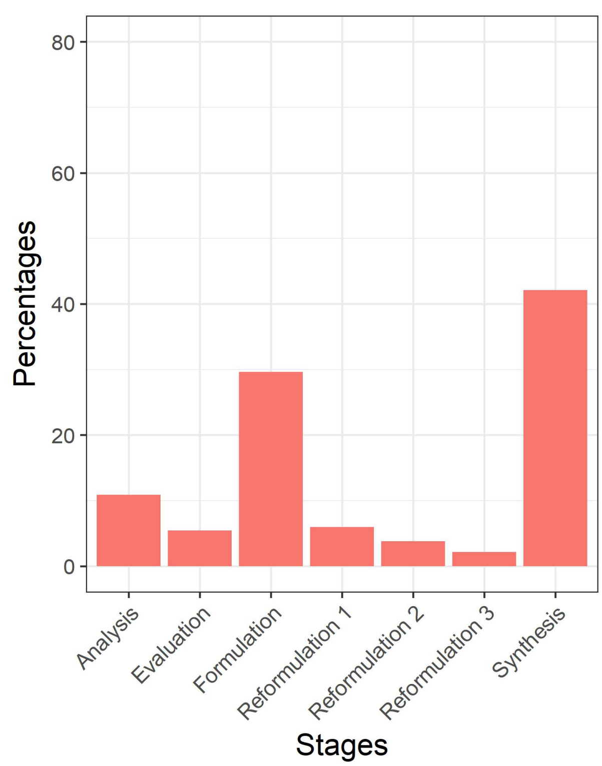


Figure 5.2: Design process stages distribution of designer A10

the transitions from Structure to Expected Behavior; and Re formulation 3 is the process from Structure to Function. Documentation (D) is the description of the whole process. Motivated by the second research question, in addition to using Markov chain to study the sequential patterns in design process, we also investigate the design process in frequency domain, i.e., how frequent each of the design stages is utilized by designers in the whole design process. An example of one designers distribution of design process stages using FBS model is shown in Figure 5.2. It indicates his/her design utility consists of Formulation and Synthesis. Both the transition probability matrix of Markov chain and the distribution of design process stages can be converted to vectors that quantify the features of design behaviors, from which different clustering methods can be applied. For example, the $N \times N$ transition probability matrix generated from first-order Markov chain of one designer can be converted to a $N^2 \times 1$ vector, and a designers design stages distribution can be converted to a $N \times 1$ vector, where N is the number of stages in a design process model. For n designers, respective $N^2 \times 1$ or $N \times n$ matrices will be formed. In this paper, we perform clustering methods on both matrices to analyze designers sequential decision-making in both time domain and frequency domain.

5.2.3 Clustering Methods

The goal of a clustering method is to divide data into a meaningful and useful groups based on their similarities [120]. In the field of engineering design, clustering has been used in many applications and various clustering methods e.g., partition-based clustering [121], shape-based clustering [122], hierarchical clustering [123], density-based clustering [124], network-based clustering [125], etc. have been adopted. In this paper, we adopt K-means, hierarchical, and network-based clustering methods to study the differences and similarities of designers sequential design behaviors. These three different methods are chosen as representatives from three different categories of clustering: hard clustering, flat clustering and network clustering [126], that covers most commonly used clustering methods. A brief description of each method is summarized as follows.

K-means clustering

K-means is one of the most popular clustering methods for partitioning dataset into distinct, non-overlapping clusters. The goal is to partition the dataset into K clusters such that the total within-cluster variation, summed over all the clusters, is minimum [127]. There are many ways to define the within-cluster variation. The most commonly used method is squared Euclidean distance. Since K-means requires the number of clusters as input, a separate algorithm is often

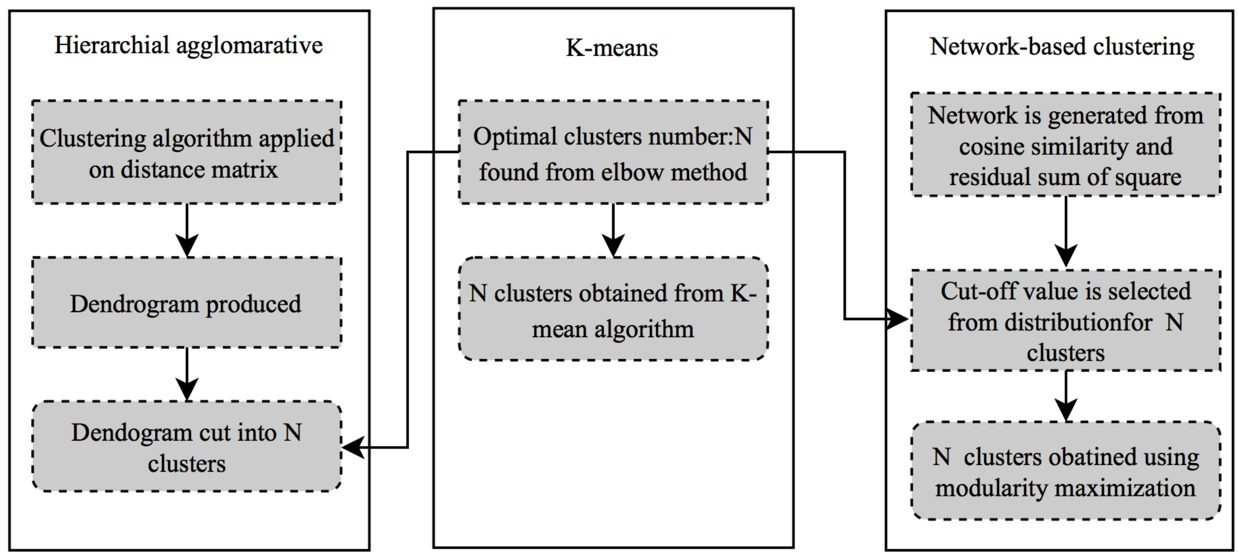


Figure 5.3: Procedure of clustering methods and the selection of optimal clustering numbers for cross comparison

needed to determine the optimal number of clusters. In this paper, the elbow plot method [117] is used to help make decisions on choosing the number of clusters. For fair comparison across different methods, the number of clusters obtained from the elbow plot method is also used to guide the implementation of the other two clustering methods (see Figure 5.3), introduced as follows.

Hierarchical clustering

Different from K-means clustering method, hierarchical clustering does not require the numbers of cluster as input initially. Instead, it produces a tree-based representation of the observation, called dendrogram. In this study, we adopt the commonly used agglomerative clustering algorithm [128] to generate this dendrogram. This algorithm starts with considering each data point as an individual cluster. The two clusters that are most similar to each other conjugate to one cluster. This process iterates and not stop until all the observations create one group and complete the dendrogram. After the dendrogram is obtained, researchers can cut it into the desired number of clusters.

Network-based clustering

In addition to the two clustering methods above, we also develop a network-based clustering approach based on network community detection technique[125]. In this method, a similarity

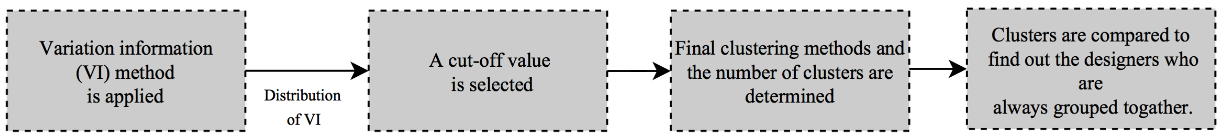


Figure 5.4: Verification of clustering results using variation of information

network of designers is first constructed, in which, nodes represent designers and edges represent the similarity between designers. In this study, the residual sum of squares (RSS) [128] and cosine similarity (CS) [128] are used as the similarity metrics. The RSS calculates the sum of the squared differences between the behavioral vectors of two designers, and CS returns the cosine angle between two vectors. Based on their measurement, a similarity matrix can be generated and its elements indicate the similarity between every pairs of designers. In order to retain the strong similarities only, a threshold value is selected to binarize the similarity network. Once the network is ready, different network community detection algorithms can be applied. We utilize the most popular and robust method [129], modularity maximization algorithm [130] to cluster the network. Since the algorithm will automatically cluster the network into an optimized number of clusters, no predetermined number of clusters is needed. To enable the comparison between the three clustering methods, we trial and error the threshold value of similarities (i.e., the RSS and the CS values) until the number of clusters in the network matches the one obtained from K-means elbow plot. See Figure 5.3 for the whole process and the connection between the network-based and K-means clustering methods

5.2.4 Verification & Validation

Since each clustering method produces its own cluster results, for verification purpose, we compare the clustering methods to verify the results. VI is an information-theoretical type of measurement which has been recently found very useful when comparing clustering methods [131]. VI measures the information lost and gained when it changes from one cluster to another. The lower a VI value is, the better is the partial agreement between two cluster. After obtaining the VI values for each pair of the clustering methods, the methods that have larger partial agreement can be identified, and the designers who have been always grouped together can be found and similar behavioral patterns can be mined from the data. Figure 5.4 shows the entire procedure. In the following sections, we apply our approach to cluster designers sequential decision-making behaviors in a solar energy systems design project.

5.3 Clustering Design Behaviors in Solar Energy System Design - A Case Study

In this section, we first give a brief description of the design problem. Next, we introduce our experiment procedure for data collection and finally, we present the collected data and FBS-based coding scheme.

5.3.1 The Design problem

The design problem in this case study is to build a solarized energy-plus home for a client in Dallas. See an illustrative example in Figure 5.5. The design objective is to maximize the annual net energy (ANE). The budget for the house is $200,000. The house should have a minimum height of 2.5 m, and the roof must be pitched. The building needs to have at least four windows and one door. The solar panel must be placed on the roof. The other constraints are shown in Table 5.1.

This design is a system design problem that involves many components (e.g., windows, roof, solar panel, etc.), many design variables (e.g., the number of solar panels, the cell efficiency of solar panel, etc.), and complex coupling relations among the variables. Therefore, the design space is very large. This is why the requirements and the constraints are developed to reduce designers action space to a manageable level. During the design process, designers make trade-off decisions. For example, there is no restriction on the area of the house. But if the area is too small, designers will not be able to place enough solar panel on the roof. As a result, the ANE will be insignificant. On the other hand, if the area is too large, the cost may exceed the budget. So, designers follow their own strategies during the design process to sequentially make decisions guiding the exploration and exploitation of the design space so as to improve the ANE as much as possible.

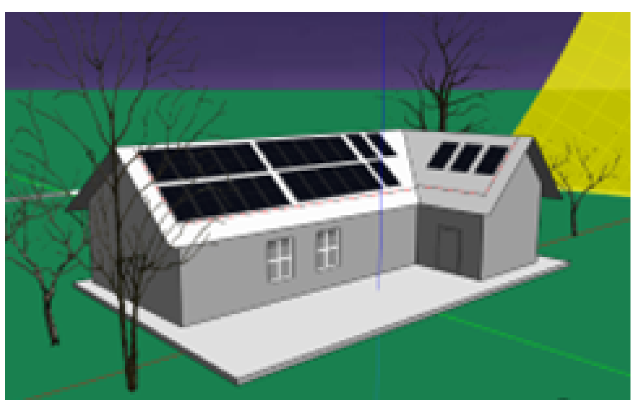


Figure 5.5: An illustrative example of the energy-plus home design project

5.3.2 Experiment procedure

In order to collect the design action data, a human-subject field experiment [132] is conducted. The energy-plus home design project is performed based on Energy3D a full-fledged computer-aided design (CAD) tool for solar energy systems [?]. Energy3D has built-in modules of engineering analysis, science simulation and financial evaluation. This ensures the collection of inter-stage design iteration data, e.g., how designers make decisions on a scientific basis (e.g., the ANE analysis results) and economic considerations (e.g., the overall cost), without disrupting the design process and designers thoughts. Energy3D can automatically log and sort all user actions, at an extremely fine grained level. All these features enable us to collect high-fidelity data which reflects designers rational behaviors.

Total 38 people, including both students and faculty members from the University of Arkansas participated in the experiment in five sessions. The participants come from different departments, and the demographics is diverse1. The participants are indexed based on which session they were in and which laptop they used. We use letters A to E for the session names, thus A02 means the participant was in Session A and sit in laptop #2.

Each session consists of two phases: pre-session and in-session. The pre-session is thirty minutes long and allotted for participants to practice Energy3D. The design of this pre-session is to account for the learning curve of humans. The data generated in pre-session was not be used for analysis. To further mitigate the learning effects, a tutorial on key operations and terminologies of Energy3D is provided to make sure the participants are familiar with the software environment2. At the end of the pre-session the participants will be guided to transition to in-session phase. In-

Table 5.1: The design requirements for energy-plus home

Item	Requirements
Story	1
Roof style	Pitched
Number of windows	≥ 4
Size of windows	$\geq 1.44\ m^2$
Number of doors	≥ 1
Size of door (Width × Height)	1.2 m × 1.2 m
Height of wall	≥ 2.5 m
Solar panel placement	Only roof
Distance between ridge to solar panel	≥ 0

session phase lasts about one hour and half. The design statement and the design requirements are provided at the beginning of this session. A record sheet is provided for participants to record the ANE and cost whenever they iterate their designs. Monetary rewards are provided at the end of the session to incentivize the participants to explore and exploit the design space as much as possible. The participants are rewarded based on the amount of time they have spent as well as the quality of their final design outcomes, which are related to the ANE and construction cost.

5.3.3 Data collection and the FBS-based coding scheme

Energy3D logs every performed action and intermediate artifacts (as Energy3D files) every 20 seconds [39]. In our experiment, 220 intermediate files are collected on average and the action log file contains on average 1500 lines of data. The action log file is saved in JSON format and includes time-stamps, design action and its corresponding parameters and/or analysis values, such as the coordinate of an object and/or ANE output. See an example as below:

{"Timestamp": "2017-11-14 12:51:27", "File": "EnergyPlusHome.ng3","Add Rack": {Type: Rack, Building: 2, ID: 23, Coordinates: [{x: -28.863, y: -49.8, z:20.799}]}

In this study, only the design actions, e.g., Add wall, Edit wall, Show heliodon, etc. are extracted for analysis. Trivial actions that do not affect the design quality, such as Camera, Add human, Edit human etc., are ignored. The participants have tried 115 different types of actions. After collecting the action data, we develop a coding scheme (Table 3) based on the FBS-based design process model to transform the design action data to the design process data in support of the cluster analysis.

Table 5.2: The FBS model and the proposed coding scheme for design actions

Design process	Definition and interpretation	Types of design action
Formulation	Generate Function from Requirement and from Function to Expected Behavior.	Add any components
Analysis	The process is generated from Structure.	Analysis of annual net energy
Synthesis	Generate and tune Structure based on the Expected Behavior.	Edit any components
Evaluation	The comparison between the Expected Behavior and the behavior enabled by the actual structure.	Cost analysis

Table 5.2 (Cont.)

Design process	Definition and interpretation	Types of design action
Reformulation 1	The transition from one structure to a different structure.	Remove structure
Reformulation 2	The transitions from Structure to Expected Behavior.	Remove solar device
Reformulation 3	The transition from Structure to Function.	Remove other components

In FBS ontology, Formulation is the process to generate Function from requirement. In our design problem, with the provided design requirements, designers start to generate house functions by adding new components, e.g., wall and window. So, we define these actions as Formulation. In Energy3D, to increase solar energy (i.e., the expected behavior), modification of different fictional components is required. So, Synthesis in our context corresponds to editing actions, e.g., change height, edit wall, etc. Analysis indicates the process of generating behavior from structure. In Energy 3D, such a process refers to ANE analysis of a given house structure. During the design, designers evaluate the overall design quality by comparing the ANE per dollar cost of different design alternatives. Therefore, give the same ANE, the action of doing cost analysis indicates the Evaluation process. Finally, in Energy3D, designers recreate structure by removing old structural components. Solar panels are sometimes removed to more precisely adjust the roof space in order to put more solar panels so as to produce more solar energy. Therefore, in our design problem, Reformulation 1, 2 and 3 refers to removing structure, solar devices, and other miscellaneous components (e.g., roof, tree, etc.), respectively. A complete coding scheme for this study in shown in Table 5.2.

5.4 Result and Discussion

5.4.1 Clustering sequential decision making based on Markov chain model

To quantify designers sequential decision-making behaviors, the first order Markov chain transition probability [69] is calculated. An entry of the matrix $_{ij}$ defines the probability that design process i transitions to j, which is calculated by the following equation,

$$\pi_{ij} = \frac{n_{ij}}{n_i} \tag{5.1}$$

,where n_{ij} is the number of times design process j is the followed by process i. n_i is the total counts of the process i during the entire design.

As an example, Figure 5.6 shows the transition probability of designer C14. It shows that the most occurred transition is Reformulation 1 → Reformulation 1 and the value is 0.75. This

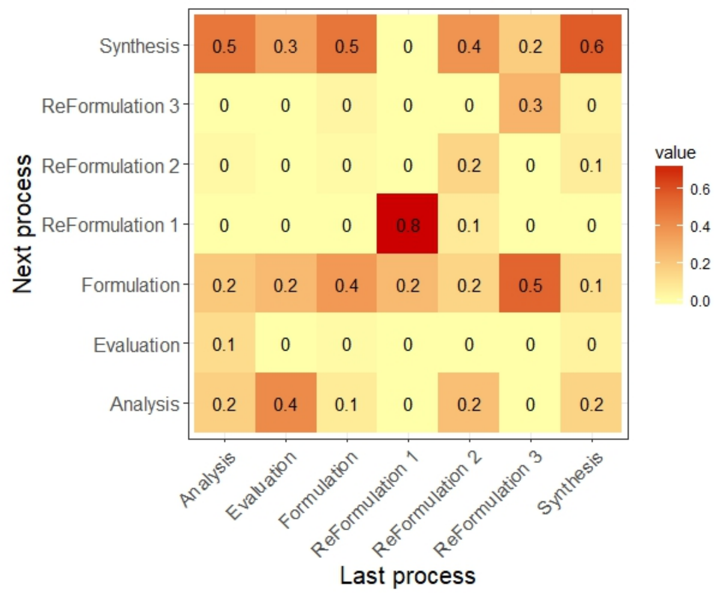
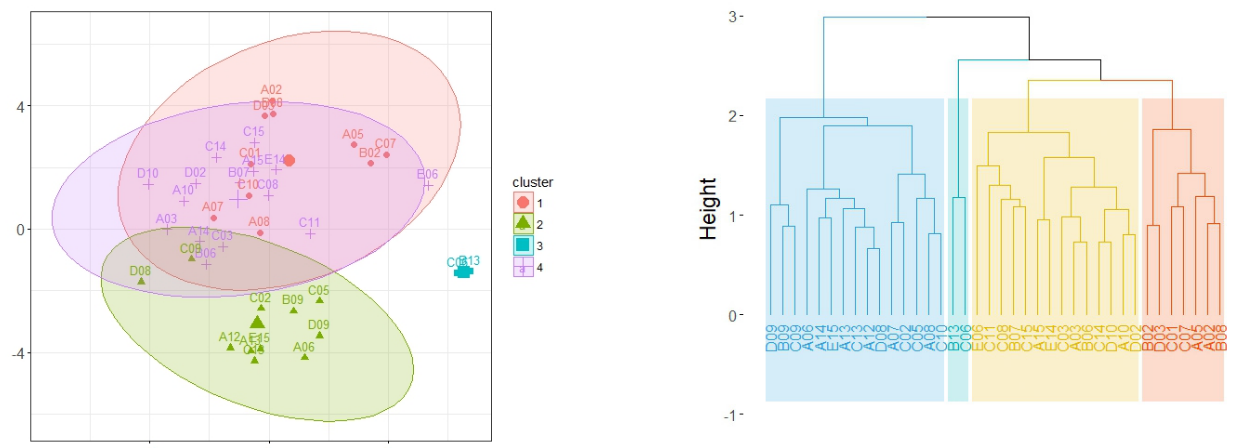


Figure 5.6: Transition matrix of the first-order Markov chain for participant C14

indicates that the designer C14 was involved in removing structure (wall, window) significantly more frequent than other transitions. The value zero means that the designer never used that transition in the design. For example, the value from Synthesis to Reformulation 1 is zero. This indicates that after editing or changing the parameters of any structural components (such as walls), this designer would never removed those components. Once all the 38 participants transition probability matrices are obtained, they are converted to a 49×38 matrix that captures the sequential design process features, from which different clustering methods are applied. The optimal numbers of clusters for K-means clustering are 4, 5 and 6, which is obtained from the elbow plot technique. This means these three points correspond to the transition region where the change of the slope on the elbow plot curve is the largest. In this paper, we evaluate different clustering methods at each of the three clustering settings. Figure 5.7(a) shows the K-means clustering results with 4 groups. The clusters are indicated by four different symbols (1, 2, 3, and 4). The number of designers in each cluster is 15, 11, 10 and 2, respectively. The plot shows the data points in two principle dimensions. From the figure, it is observed that designers B13 and C06 in Cluster 3 are situated far from the other clusters in the Euclidean space. It is inferred that their sequential behaviors are quite different from the other designers.

Hierarchical method clusters the designers by forming a dendrogram, as shown in Figure 5.7(b). The height of the dendrogram indicates the designers behavioral similarity. To get 4 clusters, the dendrogram is cut at the height of 2.1. The resulting clusters contains 15, 14, 9 and 2 members,



(a) K-means clustering plot of four groups

(b) Dendrogram produced by hierarchical agglomerative algorithm

Figure 5.7: K-means and Hierarchical clustering

respectively. Figure 5.7(b) indicates that designer A12 and D08 meet at the lowest distance (the lowest height) on the dendrogram than any other pairs. Therefore, they share the most similarity in sequential behaviors. Like K-means-4 clustering, Hierarchical-4 (HAC-4) clustering proves the similarity between B13 and C06 as well. While in K-means-4 clustering, A10 and A14 are in the same group, but in the HAC-4 clustering they are located at two different groups. This reveals that the inconsistency among different clustering methods.

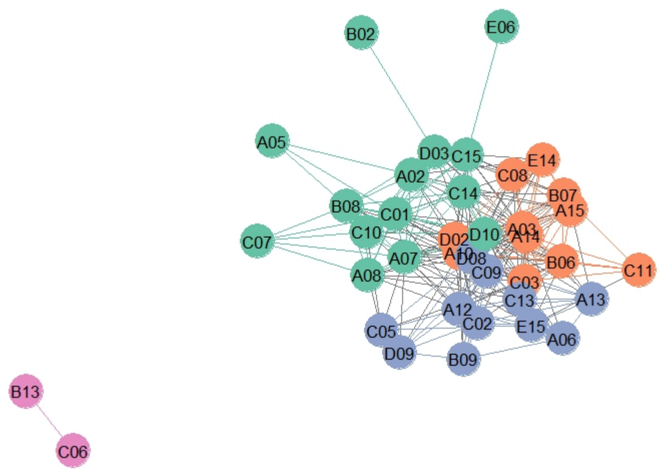


Figure 5.8: The network-based clustering using residual sum of square similarity groups the designers in four clusters

Table 5.3: Comparison of different clustering methods using variation of information. The row and column names indicate the cluster method and corresponding number of clusters

	KM-4	KM-5	KM-6	HAC-4	HAC-5	HAC-6	RSS-4	RSS-5	RSS-6	CS-4	CS-5	CS-6
KM-4	-	-	-	0.546	0.687	0.710	0.449	1.200	0.946	0.760	0.273	0.310
KM-5	-	-	-	0.894	0.815	0.752	0.785	0.847	0.594	0.919	0.596	0.633
KM-6	-	-	-	1.250	1.170	1.170	0.985	0.737	0.886	0.862	0.936	0.955
HAC-4	0.546	0.894	1.250	-	-	-	0.978	1.469	1.130	1.250	0.696	0.733
HAC-5	0.687	0.815	1.170	-	-	-	1.120	1.389	1.050	1.392	0.820	0.857
HAC-6	0.710	0.752	1.107	-	-	-	1.142	1.334	0.836	1.414	0.757	0.794
RSS-4	0.449	0.785	0.985	0.978	1.120	1.142	-	-	-	0.953	0.821	1.331
RSS-5	1.200	0.847	0.737	1.469	1.389	1.334	-	-	-	0.647	1.419	0.673
RSS-6	0.946	0.594	0.886	1.130	1.050	0.836	-	-	-	0.684	1.419	0.710
CS-4	0.760	0.919	0.862	1.250	1.392	1.414	0.953	0.647	0.684	-	-	-
CS-5	0.273	0.596	0.936	0.696	0.820	0.757	0.821	1.419	1.419	-	-	-
CS-6	0.310	0.633	0.955	0.733	0.857	0.794	1.331	0.673	0.710	-	-	-
Efficiency	5	3	0	2	1	0	1	2	2	2	3	

For the network-based clustering, we calculate the RSS and CS similarities between each pair of designers using the vectors obtained from the transition probability matrix. This process produces two 38×38 similarity matrices from which the RSS-based network and the CS-based network can be obtained, respectively. To obtain the desired number of clusters (i.e., 4, 5 and 6 determined by elbow plot method), we trial and error the RSS and CS values together with the modularity-maximization algorithm to determine the threshold. The results suggest that the values 1.24, 1.23 and 1.22 of RSS similarity are able to create 4, 5 and 6 clusters, respectively for RSS-based network. In the CS-based network, it is found that the values of 0.7, 0.75 and 0.77, are the appropriate threshold values to produce the desired number of clusters 4, 5, and 6.

Figure 5.8 shows the result of RSS-based network clustered in 4 groups indicated by different colors. The four groups consist of 14, 11, 11 and 2 members, respectively. But in this method, the clustering results are different from K-means-4 and HAC-4. For example, E06 and E14 belong to the same group in K-means-4 and HAC-4, but in RSS-4, they are in separate groups. But results from different methods do hold consistency. For example, B13 and C06 have been always grouped together in all three methods. Following the same approach of generating RSS-based network clustering, clusters can also be produced using CS-based network clustering method. CS-based clustering shows some similarities and dissimilarities as well. For example, B13 and C06 are clustered together with K-means-4, HAC-4 and RSS-4 methods, but they are separated with CS-6 method.

Since clustering results are inconstant from different clustering methods, the results need to be verified. The variation of information (VI) is used to compare each pair of clustering methods to evaluate the partial agreement between the clusters obtained from each method. The VI values are summarized in Table 5.3. Please note that the VI between the same clustering methods but different cluster numbers (e.g. K-means-4 vs. Kmeans-5) is not worth comparing, thus the corresponding VI are not available in Table 5.3. From Table 5.3, we can observe that the VI between K-means-4 clustering and CS-6 clustering is 0.31. On the other hand, the VI between HAC-5 and CS-5 clustering have is 0.82. So, K-means-4 clustering and CS-6 clustering have more overlapping cluster members than that HAC-5 and CS-5 clustering has.

By analyzing the distribution of VI (see Figure 5.9), the value of 0.7 (corresponding to the top 25 % quantile) is chosen as a cutoff value to filter out the clustering methods that have more consistent results. During this process, we are able to a) find the most efficient clustering method and its corresponding number of consideration. The values which are below 0.7 are considered as efficient. This can be expressed as the following way:

$$Efficiency = \sum_{i}^{k} f(VI_i) \tag{5.2}$$

, where $f(VI_i) = 1$ if $VI_i < 0.7$; and 0 otherwise. i= 1, 2.... k and k = 12 in this case study. It is observed that K-means-4 clustering has the largest number of times in overlapping with other clustering methods. Therefore, K-means-4 clustering is the most efficient method among all the three methods in consideration. Detailed results of K-means-4 clustering is presented earlier. By checking the occurrence of the VI values being below the threshold 0.7, we identify K-means (4, 5), HAC (4, 5), RSS-(4, 5, 6) and CS-(5, 6) for consideration to identify the designers who have been always clustered together irrespective to the methods being used. The results are shown in Table 5.4. Note that each row of designers are grouped together without any pre-knowledge.

With the clustering results, we revisit the first question we aim to answer in this study: What are the most frequent sequential design behavioral patterns that most designers would follow in systems design? By analyzing the clusters, it found that, for most of the cases, the highest transition probability for each designer in a group is similar. The sequential design behaviors that most designers follow are listed and discussed below:

- Synthesis $\rightarrow$ Synthesis

  This transition of design stages is the most frequently occurred pattern. For example, the highest transition probability of all the designers of the third group (A06, A12, A13, C13, D08, and E15) is Synthesis Synthesis. Again, the fifth group (B11, C06) also uses this pattern very often. It indicates that the designers of these groups kept modifying the parameters of the components. The possible reason for this deign pattern is that designers are incentivized

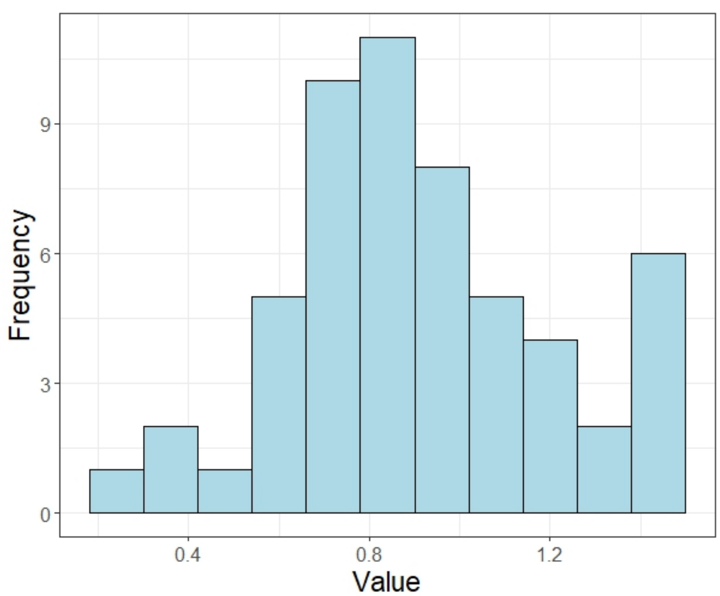


Figure 5.9: Distribution of the VI shown in Table

by the rewarding mechanism in the experiment, thus they tried their best to exploit the design space by sequentially changing the design parameters.

- Reformulation → Formulation

  Designers also used this pattern very frequently. We found that, the highest transition probability of the second group (A03, A15, B07, C08, D02, D10, and E14) is Reformulation 2 → Reformulation. This pattern indicates that designers in this group spent significant amount of time to remove solar panels and again adding them back. It may be due to that they were trying to adjust the solar panel on the roof to a perfect condition. Again, the last group (B09, D09) followed the Reformulation 3 → Formulation design pattern. Designers in this group spent most of the time to remove the existing roof or others component (excluding solar panels and structural components) and again adding it.

5.4.2 Clustering design behaviors based on the distribution of design process stages

The second question wed like to answer is that, If designers behave similarly in sequential design-making of time domain, would they also have similar behaviors in frequency domain? To answer this question, we apply the same approach in Figure 5.1 to identify the designers who use similar number of design process stages during their designs. The only difference between this analysis and the one in the previous section is that the behavioral data used in this section in a 7×1 vector. Each element of this vector is the frequency of each design process stage. Therefore, this analysis capture similarities of designers who have similar preferences of leveraging certain design processes in systems design. Figure 5.10 shows the examples of the distributions of the design process stages from four designers. These results verify that our approach is able to successfully

Table 5.4: Designers who are always in the same groups in sequential clustering

A02, A05, B08, C01, C07
A03, A15, B07,C08, D02, D10,E14
A06, A12, A13, C13, D08, E15
A07, A08, C10
B06, C11
B09, D09

cluster the similar design behaviors together.

Table 5.5 shows the designers who have been always grouped together based on their design process distribution irrespective of the clustering methods. Among all the participants, it is found that Synthesis is the most frequently used design process stage. Out of the 10 similar behavioral groups, 9 groups follow this trend. That means, in their design processes most of the time they are involved in editing various component of the energy-plus home. Such a behavior is again a reflection of the reward incentive created in the experiment. However, as shown in Figure 11, B08 and B09 do not follow this trend. Instead, the most frequent design process stage is Formulation which signifies that their design was much involved adding components to meet the design requirements. Participants B13 and E06 have a unique distribution of the design process stages. Instead of using all seven design processes, they are mainly involved in Formulation, Synthesis, Analysis, and Evaluation process. They almost never performed any actions related to reformulation. This indicates that different designers have different patterns and designers do have preferences in selecting certain types of design actions to explore the design space. The resulting distribution of design actions is therefore not uniform. It is observed that Reformulation is overall used less frequently than other process stages on average. This implies that designers are incline to improving the design quality by editing the artifacts that are already established rather than removing and restructuring the house. Some of the designers (e.g., A05, C05) performed Analysis almost the same number of

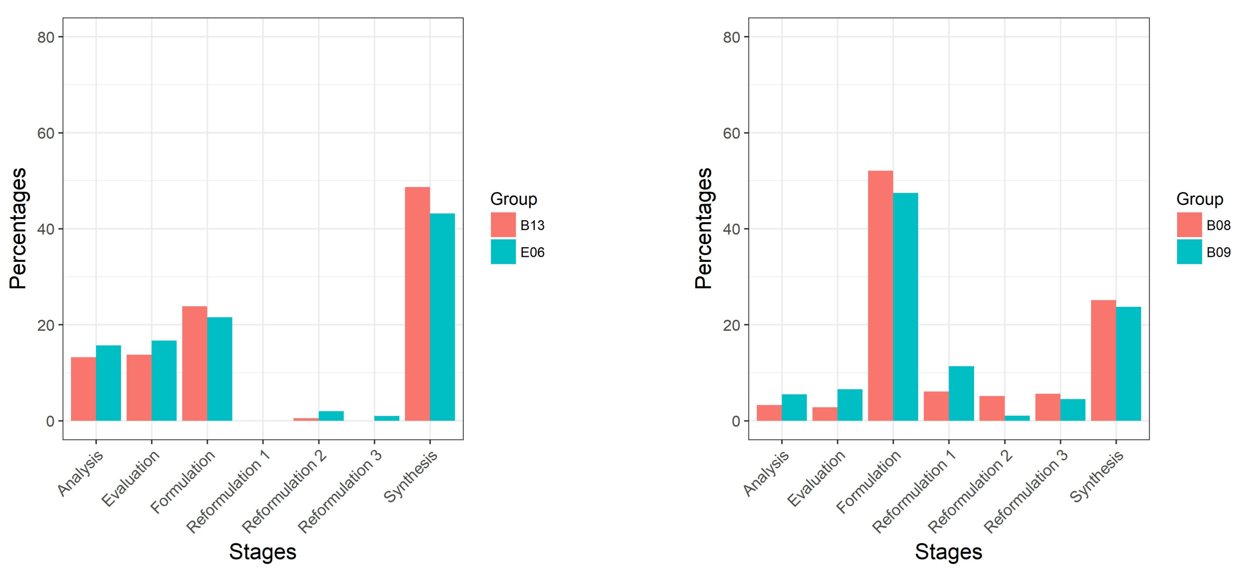


Figure 5.10: Design process stage distribution of two groups where designers in the same group show similar patterns of distribution whereas the behavioral patterns are different between groups

Table 5.5: Clustering results of design process distribution irrespective of the clustering methods

A02, A14, C15, D02
B02, C01
A06, B06, B07, C02, C10, C13,D03
A10, D10
A12, E15
A05, C05
A08, C09, C11
B08, B09
A13, C14
B13,E06

times as Synthesis and Formulation. This behavior indicates that they were exploring the effects of changing certain parameters because any changes made in Energy3D can be immediately assessed.

By comparing Table 5.4 and Table 5.5 it is found that only A12 and E15 grouped together in both sequential behavioral analysis and distribution analysis of design process stages. This indicates that, for most designers, even if they behave similarly in sequential design-making of time domain, they do not necessarily have similar behaviors in frequency domain.

5.5 Conclusion

This chapter presents a framework of automatically clustering designers with similar design behaviors. Fine-grained design action data are collected using Energy3D in an non-intrusive way. Then, the first-order Markov chain is used to generate the sequential behavioral data after applying the FBS-based coding scheme. On the other hand, based on the distribution of design process stages, we analyzed the designers behaviors quantified in frequency domain. We utilized three representative clustering methods, K-means, the hierarchical agglomerative, and the network-based clustering methods in this study. The elbow plot method indicates that 4, 5 and 6 are preferred clustering numbers. In order to verify the clustering results, variation of information method is used and we find that K-means with 4 clusters is the most efficient clustering method. Finally, by comparing the obtained clusters, designers with similar sequential behavioral patterns are identified.

We find that, Synthesis $\rightarrow$ Synthesis and Reformulation $\rightarrow$ Formulation are the design patterns that were followed by a large number of designers. In addition, we find that designers who used the same number of process stages do not necessarily follow the same sequence in their design.

The overall contribution of this study is the development of a general framework that can accommodate various clustering methods for identifying design behavioral patterns. Moreover, the network-based clustering approach developed in this study provides a new way for clustering design behaviors by leveraging network community-detection algorithms. Successful identification of similar behaviors as well as their design patterns has significant benefits in discovering efficient design heuristics and guiding team-based design. For example, useful design process stage frequencies and design patterns that lead to better design outcomes can be identified by correlating design quality with different behavioral groups. Also, in team-based design, to maximize the working efficiency, similar/dissimilar designers could be paired up to improve the communication and /or diversity within a group.

6 Modeling Design Behavior From Multiple Dimensions

6.1 Overview

In the previous chapter, we model one-step sequential behavior using first-order Markov chain model. This motivates us to identify design behaviors from multiple dimensions. In this chapter, we represent design thinking as an intermediate layer between human designers thought processes and their design behaviors. To do so, this paper first identifies five design behaviors based on the current design theories. These behaviors include design action preference, one-step sequential behavior, contextual behavior, long-term sequential behavior, and reflective thinking behavior. Next, we develop computational methods to characterize each of the design behaviors. Particularly, we use design action distribution, first-order Markov chain, Doc2Vec, bi-directional LSTM autoencoder, and time gap distribution to characterize the five design behaviors. The characterization of the design behaviors through embedding techniques is essentially a latent representation of the design thinking, and we refer to it as design embeddings. After obtaining the embedding, an X-mean clustering algorithm is adopted to each of the embeddings to cluster designers. The approach is applied to data collected from a high school solar system design challenge. The clustering results show that designers follow several design patterns according to the corresponding behavior, which corroborates the effectiveness of using design embedding for design behavior clustering

6.2 Technical Background and Research approach

In this section, we first briefly introduce the research approach adopted in this study. Next, we present the technical background for different embedding techniques.

6.2.1 Theoretical background

One of the major contributions of this study is the identification of five design behaviors for studying design thinking representation. Therefore, before describing the overall research approach, we would like to present the rationale of how the five behaviors are identified. These behaviors include one-step sequential behavior, contextual behavior, long-term sequential behavior, reflective thinking behavior, and action preference.

The one-step sequential behavior, contextual behavior, and long-term sequential behavior are selected based on the mental iteration model [66]. Design is a goal-directed problem-solving process and can be modeled as an iterative and sequential decision-making process. Jin and Chuslip

[66] proposed a cognitive model to describe the mental iteration during design. According to that model, in every design process, several cognitive activities occur, such as generate, compose, evaluate, etc. Also, different iteration loops are embedded in the design process. These loops collectively generate a global loop. Besides the global loop, each cognitive activity defines a local loop. In complex systems design problems, these loops frequently occur as designers go back and forth iteratively between different stages to search the design space and take different design actions to accomplish required design tasks. Therefore, in this study, we propose to use one-step sequential behavior and contextual behavior (short-term behaviors) to capture the local-loop behavioral patterns and use long-term sequential behaviors to capture the global-loop iterative patterns.

Next, we consider reflective thinking. The core of reflective thinking is metacognition and self-monitoring, which help designers to reflect experience and knowledge in their actions as well as provide feedback to improve the design process [133]. In a design process, designers may take various modes of reflective thinking. For example, some designers use a bigger picture (take a longer time to think) while others use a micro-scoping view (take a shorter time to think). Reflective thinking behavior enables designers to scrutinize their thinking, behavior, design process, thus produce higher quality designs [134, 135]. Therefore, understanding and computationally modeling designer reflective thinking are important.

Lastly, we study designers action preferences based on how frequently a designer uses different types of design actions (i.e., the distribution of design actions) during a design process. In total, five different design behaviors are adopted from three dimensions mental iteration, reflective thinking, and design action preferences. We envision that modeling the design behaviors from multiple dimensions can help better understand design thinking.

6.2.2 Research approach

The overall approach (see Figure 6.1) starts with collecting the raw design action data from different sources, such as CAD loggers, design documents, etc. This raw design action data contains design actions, design-related artifacts, and the values of various design parameters. After collecting the design action data, to computationally model these five design behaviors, we adopted five different techniques. We use design action distribution to study design action preferences, the Markov chain model to study the one-step sequential behavior, the Doc2Vec to model contextual behavior, the bi-directional LSTM autoencoder to study the long-term sequential behavior, and the time-gap distribution to analyze reflective thinking. To explain the overall process, suppose a designers sequence of design actions $[a_1, a_2, a_3, , a_N]$ which has a timestamp associated with it $[t_1, t_2, t_3, , t_N]$.

Before analyzing the design action preference and the one-step sequential behavior, we apply an ontological design process model (e.g., the FBS model), which consists of several design stages to characterize the design process. By applying the design process model, we will obtain a sequence of design process stages $[p_1, p_2, p_3, , p_N]$. With this operation, we can reduce the dimensionality of the design action data. This treatment is similar to an embedding (latent space representation), which can help interpret designers thought processes. To elicit designers action preferences, we count the total number of each design process stage that certain actions fall into and plot the resulting distribution for every designer. To understand designers one-step sequential behavior, we apply the first-order Markov chain to every designers design process stage sequence and compute the transition probability matrix. This transition probability matrix can be vectorized, which quantifies the features of the one-step sequential behavior. For example, given a design process model defining N design process stages, we can get an $N \times 1$ vector from action preference, and an $N \times N$ transition probability matrix from the Markov chain model for one designer. The transition probability matrix can be converted into an $N^2 \times 1$ vector. For n designers, two matrices in the dimension of $N \times n$ and $N^2 \times n$ can be formed, representing the aggregated action preference and the one-step sequential behavior, respectively.

To understand designers contextual behavior and long-term sequential behaviors, we apply the Doc2Vec [136] and the bi-directional LSTM auto-encoder [137] on the design action sequence, respectively. Both Doc2Vec and bi-directional LSTM attempt to predict the next design action

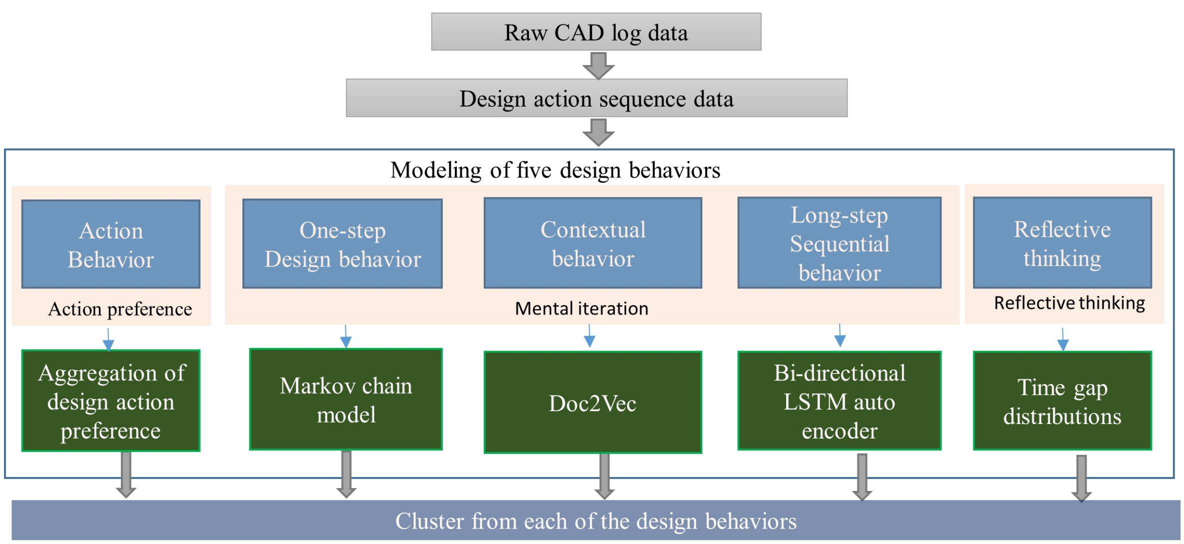


Figure 6.1: The research approach for studying design thinking based on five design behaviors

from the input sequence. Doc2Vec supports this process by training paragraph vectors as auxiliary information. We will get an embedding matrix from each of these methods. As the embedding matrix is already a representation of the relationship among design actions, the data transformation from design action to design process stage using an ontological design process model is not needed in these two methods. It is mention-worthy that the size of the embedding matrix is user-defined. For example, with the embedding size of M, and for n designers sequences, an $M \times n$ dimensional matrix from each of the methods can be obtained.

To understand the designers reflective thinking, we utilize the time-gap distribution analysis. Particularly, we consider the time gap between each design action performed by a designer. For example, for an action sequence, the time gaps are $[0, t_2 - t_1, t_3 - t_2..t_n - t_{(n-1)}]$. The distribution of this time gap essentially carries the reflective behavior. From each of the designers time gap distributions, we can get several features, such as the distribution type and its parameters. For a particular designer, we use these features to create a vector, $P = [Distname, D_1, D_2, , D_n]$, where Dist name indicates the distribution type (a categorical variable) and $D_1, D_2, , D_n$ are the distribution parameters. It is noted that the parameter number can be varied based on the type of the distribution. Assuming there are L parameters, for n designers, we obtain an $L \times n$ matrix. This matrix will be the feature representation of designers reflective design thinking behaviors.

Based on these five models, we can obtain five behavioral matrices (i.e., the design embeddings) representing the five corresponding design thinking behaviors. Then, we implement a clustering method, i.e., X-mean cluster [41], on each behavioral matrix to group the designers who have similar design behavioral patterns in different behavioral dimensions. Figure 6.1 depicts a schematic diagram of the research approaches.

6.2.3 Doc2Vec

Doc2Vec uses a neural network approach to create a fixed-length vector representation of variable length sequences, such as sentences, paragraphs. In this study, since a design action sequence is a sequence of text data, it can be treated as a sentence. Doc2Vec is based on Word2Vec, where it attempts to predict an element in a sequence from its surrounding or context element [136]. Given a sequence $w_1, w_2, w_3, , w_T$, to predict the context element w_t, the objective of the Word2vec is to maximize the average log probability.

$$\frac{1}{T} \sum_{t=kl}^{T-k} logp(w_t | w_{t-k}, ..., w_{t+k}) \tag{6.1}$$

The prediction task is typically done by a neural network architecture with a multiclass classifier such as softmax [41]. This process can be expressed as follows:

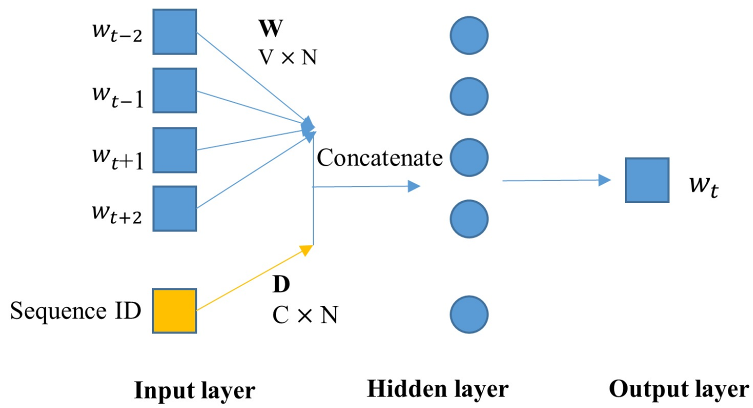


Figure 6.2: Doc2Vec architecture

$$p(w_t|w_{t-k},,w_{t+k}) = \frac{e^y w_t}{\sum_i e^{y_i}} \tag{6.2}$$

$$y = \mathbf{b} + \mathbf{Uh}(w_{t-k}, ..., w_{t+k}; \mathbf{W}) \tag{6.3}$$

, where Equation (2) outputs the predicated probability using the softmax function. y_i is the log probability for each output element i. Equation (3) represents the equation of feed-forward neural network where ***U,b*** are the parameters of neural networks. ***h*** is constructed by a concatenation of vectors extracted from **W**.

In Doc2Vec, every sequence is associated with a unique vector, which is represented by a matrix D (for all sequences, it creates a matrix). Every element of the sequence is also mapped to a unique vector which is represented as **W** in Figure 6.2. The matrix **D** and **W** are concatenated and used in Equation (3) in place of ***h***.

6.2.4 Bi-directional LSTM auto-encoder

The aim of using an auto-encoder (AE) is to learn a compressed, distributed representation of a data set. It is a neural network model that captures the most salient features of the input data [138]. The basic AE consists of only one hidden layer, and the target value is set equal to the input value. The training of the AE is done in two phases: encoding and decoding. In the encoding phase, input data are mapped into the hidden layer, and in the decoding process, the input data are reconstructed from the hidden layer representation. Given an input dataset $X = x_1, x_2, x_3, , x_n$, the two phases can be expressed as follows:

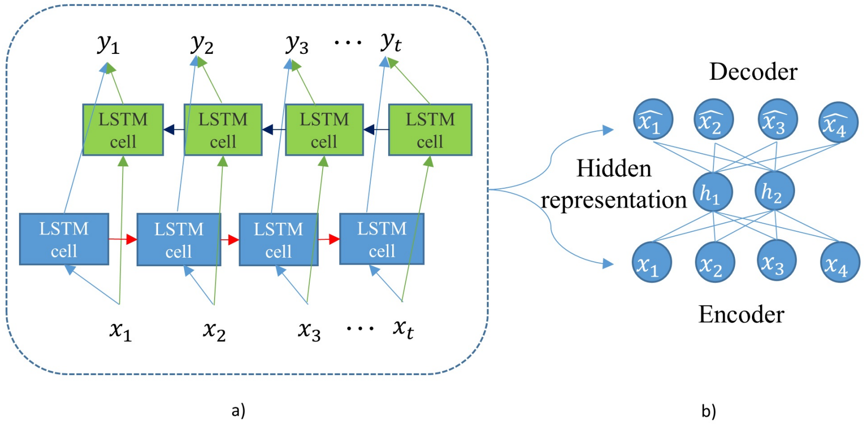


Figure 6.3: a) Bi-directional LSTM b) Basic auto-encoder

$$\boldsymbol{h(x) = f(W_1 X + b_1)} \tag{6.4}$$

$$\boldsymbol{\hat{X} = g(W_2(x) + b_2)} \tag{6.5}$$

,where, ***h(x)*** represents the hidden representations of the input vector $\boldsymbol{X}$, and $\boldsymbol{X}$ is the decoder vector of the output layer. f is the encoding function, while g is the decoding function. W_1 and W_2 are the weight matrix of the encoder and decoder, respectively. b_1 and b_2 are the bias vector in each phase, respectively. A schematic diagram of the auto-encoder is shown in figure 6.3 (b).

LSTM is an upgraded variation of the recurrent neural network (RNN) [139], which is basically a recursive neural network used for sequential data. LSTM uses a gating mechanism that solves several flaws of the RNN (i.e., vanishing gradient problem, long-term dependency, etc.). In this study, we leverage bidirectional LSTM in the auto-encoder architecture. Compared to the basic LSTM model, bidirectional LSTM consists of two groups of hidden layers. One layer for input sequence in the forward direction and the other layer for input sequence in the backward direction (see figure 6.3(a)). These two hidden layers do not interact with each other, and their output is concatenated to the final output layer. The mathematical equations for the bidirectional LSTM are the same as a basic LSTM, except that there are two hidden states at $t^t h$ time steps: $\overrightarrow{\mathrm{h}}$ (forward process) and $\overleftarrow{\mathrm{h}}$ (backward process). These two hidden states are concatenated for the final output In the AE architecture, bi-directional LSTM is replaced with the feed-forward neural network. A schematic diagram of the bi-directional LSTM autoencoder is shown in Figure 6.4.

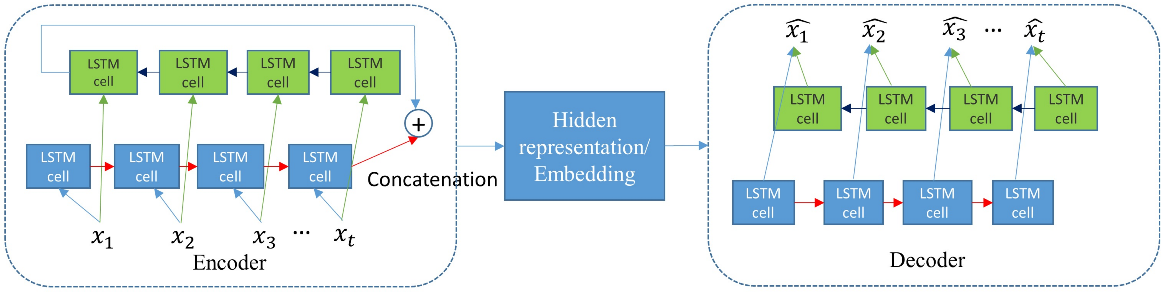


Figure 6.4: Bi-directional LSTM auto-encoder

6.3 Clustering Design Behaviors in Solar Energy System Design - A Case Study

This section provides an introduction to the design problem used in the case study and the data collection method.

6.3.1 Design procedure

The study was implemented in a suburban high school in the north-eastern US. The participants are 113 students from seven $9th$ grade classes of a course on the science of energy. These students barely had design experience before the project. During the six-day project, students worked on a design challenge with an open-source CAD software called Energy3D [111] individually and sought help from teachers if needed. Specifically, the project started with a day of Energy3D tutorial and followed by three days of conceptual learning, in which students interacted with simulations to understand five solar concepts and how these concepts affect solar-energy acceptance. Then students try to solve an authentic design challenge for two days to apply knowledge to practice and develop design skills.

6.3.2 Design problem

The five solar concepts are the Suns path, the projection effect, the effect of the air mass, the effect of weather, and solar radiation pathways. These concepts are tightly related to the design challenge and were selected by domain experts. Individual simulations and exercises were provided to students to learn each concept. The design task was customized to the students with their school as the context. The challenge was named Solarize Your School and set as asking for bids to power their school with green energy. Mainly, a 3D model of their school was provided. Students could install solar panels on the school building roof to generate no less than 400,000 kWh of electricity per year while the payback period was less than ten years. We provide three different solar panel

Figure 6.5: An example of the solarize your school design

models from which designers can choose any one of them for the design. This design challenge required students to balance several factors such as panel costs, solar panel orientation, tile angle, and avoiding shadows while aiming for the goal. Figure 6.5 shows an example of the Solarize Your School design.

6.3.3 Data collection and data processing

Energy3D collects the continuous flow of design logs, including design actions, time steps, design parameters, and simulation results. An example of a line of design action log is shown below.

{"Timestamp": "2019-10-22 08:34:26", "Project": "Stoughton High School", "File": "stoughton-high-school-ma.ng3", "Change Tilt Angle for All Racks": {"New Value": -1.0}}

Although initially, we collect 113 designers data, after analysing their design, we realize that several students did not follow the design requirements (e.g., failed to choose one of the provided solar panels). For a fair comparison, we only consider the designs that met the design constraints, and this leads to 39 valid designs.

In this study, we only collect the design action data, such as the change Tilt Angle for All Racks in the above example. By extracting the design action from every row of the log file, a design action sequence can be generated. Its worth noting that we ignore the camera-related action such as zoom in, zoom out, and camera because it does not affect the design performance per se.

Table 6.1: The coding scheme based on the FBS design process model

Design process	**Design actions**
Formulation	Add any component
Analysis	Analysis of annual net energy
Synthesis	Edit any component
Evaluation	Cost analysis
Reformulation 1	Remove structure
Reformulation 2	Remove solar device
Reformulation 3	Remove other components

After removing the irrelevant design actions, 60 unique design actions are identified. Then, for action behavior and one-step sequential behavior, we develop a coding scheme based on the FBS model to transcribe the design action data into a sequence of design processes. The coding scheme shown in Table 6.1 is used to categorize each design actions into one of the seven design process stages, including Formulation (F), Analysis (A), Synthesis (S), Evaluation (E), Reformulation 1 (R1), Reformulation 2 (R2) and Reformulation 3 (R3). The detail of the transformation process is described in prior Section 5.3.3.

6.4 Result and Discussion

6.4.1 Result

In this section, we present the result obtained from different design behaviors, particularly action preference, one-step sequential behavior, contextual behavior, long-term sequential behavior, and reflective thinking. The behaviors are represented as embedding and clustered using the X-mean clustering method. To compare the final design performance from the designers in each cluster, we developed a metric to quantify a students final design quality (DQ). This metric is as follows:

$$DQ = \frac{P_R \times B \times E_0}{P_0 \times C \times E_R} \tag{6.6}$$

where, P_B = required payback period

B = budget

E_0 = Obtained energy output

P_0 = Obtained playback period

C = Cost

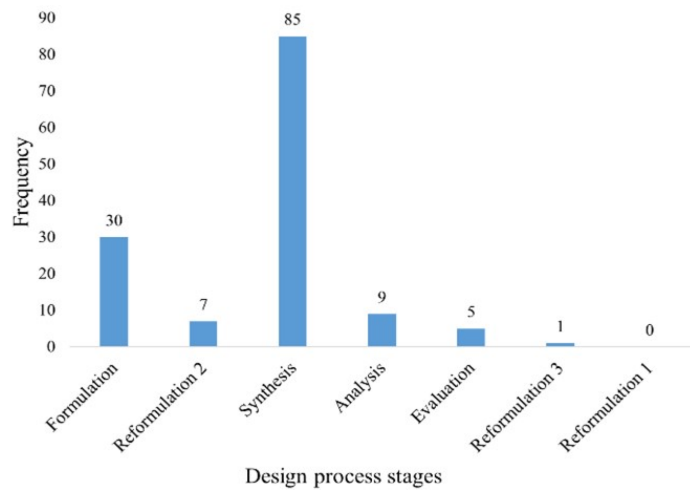


Figure 6.6: Action preferences of designer P4L25

E_R = required energy output

A students action preference is represented by the distribution of the design process stages that the student was in during the entire design process. By following the coding scheme in Table 1, we get a 7×1 vector for each designer; and for 39 designers, we get a $7times39$ action behavior matrix. Figure 6.6 shows one designers action preference distribution. By applying X-mean clustering on the action behavior matrix, three clusters are found. Cluster 3 includes ten designers who achieve the highest mean DQ of 1.325 with a standard deviation of 0.40, while Cluster 1 achieves the lowest DQ with 1.208 (standard deviation 0.408). Cluster 2 contains 13 designers with a mean DQ of 1.25 (standard deviation 0.64). Analysis of variance (ANOVA) indicates the difference between the cluster's DQ is not significant (p-value is 0.708).

Table 6.2: Clustering of one-step sequential behavior

	Cluster 1	**Cluster 2**
0	P1L10	P1L12
1	P1L14	P1L13
2	P1L17	P1L20
3	P1L18	P1L3
4	P2L10	P1L5
5	P2L12	P2L11
6	P2L13	P2L2
7	P2L14	P4L1

Table 6.2 (Cont.)

	Cluster 1	**Cluster 2**
8	P2L16	P4L10
9	P2L17	P4L25
10	P2L7	P4L28
11	P3L3	P4L32
12	P4L11	P4L5
13	P4L26	P6L12
14	P4L27	P6L17
15	P4L9	P6L18
16	P6L1	P6L3
17	P6L14	
18	P6L15	
19	P6L19	
20	P6L4	
21	P6L6	
Mean of design quality	1.26	1.25
STD of design performance	0.278	0.648

We quantify the one-step sequential behavior using the first-order Markov chain model. Particularly, the transition probability matrix obtained from the first-order Markov model is characterized as the one-step sequential behavior. Like the previous method, before applying the model, the FBS design process model transforms the design actions into the sequence design process stages. We obtain a 7×7 transition probability matrix for seven design process stages and then flatten the matrix to get a 49×1 vector. After obtaining 39 designers transition probability matrices, they are converted to a 39×49 matrix that captures the one-step design behavior, from which the X-mean clustering is applied. By clustering one-step sequential behavior, we identify two clusters. In this method, the DQ obtained from both clusters is similar. Cluster 1 contains 22 designers with a mean DQ of 1.26 (standard deviation 0.278), while Cluster 2 achieves a mean DQ of 1.25 with a standard deviation of 0.648. The t-test indicates no significant differences between the DQs of the two clusters (p-value 0.27). Table 6.2 shows the results of one-step sequential behavior clustering. Here, the designers are indicated with the class number and the laptop number. For example, P1L10 means that the designer is from Class 1 and used laptop number 10.

Using Doc2Vec, we obtain design embedding that represents the designers contextual behavior or short-term behavior. Several hyper‘ parameters need to be tuned and selected for the

Doc2Vec model. For example, in this study, we choose the embedding size for Doc2Vec as 100.

Table 6.3: Clustering of long-step sequential behavior

	Cluster 1	Cluster 2	Cluster 3
0	P1L10	P1L12	P1L3
1	P1L13	P2L14	P6L1
2	P1L14	P2L16	P6L18
3	P1L17	P2L2	
4	P1L18	P2L7	
5	P1L20	P3L3	
6	P1L5	P4L28	
7	P2L10	P4L32	
8	P2L11	P4L9	
9	P2L12	P6L12	
10	P2L13	P6L6	
11	P2L17		
12	P4L1		
13	P4L10		
14	P4L11		
15	P4L25		
16	P4L26		
17	P4L27		
18	P4L5		
19	P6L14		
20	P6L15		
21	P6L17		
22	P6L19		
23	P6L3		
24	P6L4		
Mean of design quality	1.20	1.34	1.35
STD of design performance	0.356	0.637	0.588

Additionally, we choose the context window size as 5. With these settings, for 39 designers, we obtain a 39×100 embedding matrix. We apply the X-mean clustering method on the obtained embedding matrix and get two clusters. The first cluster contains 30 designers with a mean DQ

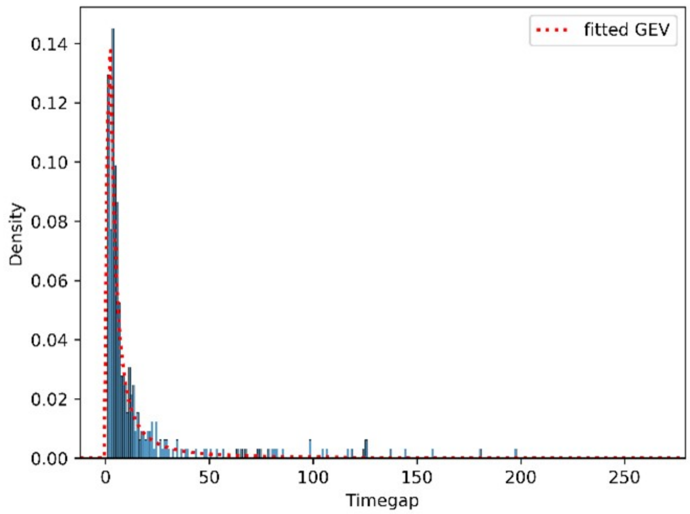


Figure 6.7: Timegap distribution of designer P1L10

of 1.22 and a standard deviation of 0.483. The second cluster contains nine students with a mean DQ of 1.34 and a standard deviation of 0.432. However, the t-test again indicates the difference between the DQs of the two clusters is not statistically significant (p-value 0.27).

We obtain design embedding for the long-term sequential behavior by utilizing the bi-directional LSTM autoencoder. In this architecture, in both the encoder and decoder parts, we use a bi-directional LSTM layer with a size of 128. Therefore, the embedding size from the LSTM autoencoder is 256, and with all the designers, we obtain a 39×256 matrix. By clustering the embedding matrix, we get three clusters. Table 6.3 shows the clustering results of the long-term sequential behavior. Cluster 1 contains 24 students with a mean DQ of 1.20 (standard deviation 0.356), while Cluster 3 has only three designers with a mean DQ of 1.35 (standard deviation 0.588). Cluster 2 contains 12 designers with a mean DQ of 1.34 (standard deviation 0.637). According to the ANOVA test, the difference among the clusters is not significant (p-value 0.7).

Finally, to obtain the embedding from reflective thinking, we get the parameters of the designers time gap distribution. We only consider the time gap between 0s to 300s. The time gap between two actions taken exceeding 300s indicates the student likely stopped the design process. So, we omit the time gaps of more than 300s. In order to understand what distribution fits these time gap distributions, we use KolmogorovSmirnov test [140], where different distributions, including Normal, Exponential, Gamma, Generalized extreme value (GEV) distribution, and Weibull distribution, are compared against. The test indicates that GEV distribution has the best fit for majority of the designers time gaps. Figure 6.7 shows designer P1L10s empirical time gap distribution and the fitted GEV distribution. From the distribution, we identify three parameters,

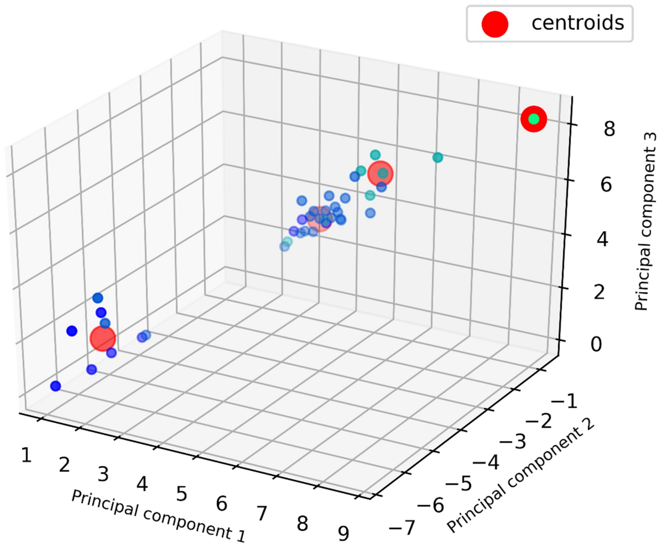


Figure 6.8: Cluster obtain from reflective thinking behavior

including shape, location, and scale. With these three parameters from 39 designers, we obtain a 3×39 embedding matrix. This matrix represents the designers reflective thinking. After applying the X-mean clustering method, we obtain four clusters. Figure 6.8 shows the four clustering results. The results of the clustering, shown in Table 5, indicate that Cluster 1 contains nine designers with a mean DQ of 1.199 and a standard deviation of 0.32. Cluster 2 contains 22 designers with a mean DQ of 1.31 and a standard deviation of 0.55, while Cluster 3 contains seven designers with a mean DQ of 1.14 and a standard deviation of 0.39. Cluster 4 has only one designer with a DQ of 1.10. The Anova test indicates that the difference in the DQs among the clusters is not significant (p-value is 0.83).

6.4.2 Discussion

This study aims to understand design thinking behaviors from different behavioral dimensions by characterizing them through design embedding. After obtaining the embedding, we apply the X-mean clustering method to each of the embedding matrices to cluster designers. The clustering results indicate that the designers are clustered not according to their final design quality but instead based on their behavioral patterns. Different design patterns in a design process can lead to similar quality of the final design. For example, in the clustering based on designers action behavior embedding, the designers of Cluster 3 use a high number of *Synthesis* on average compared to the designers in other clusters. Cluster 3 uses on average 500 *Synthesis*, while Cluster 1 and Cluster 2 use on average 150 and 233 *Synthesis*, respectively. This indicates that designers of Cluster 3 are

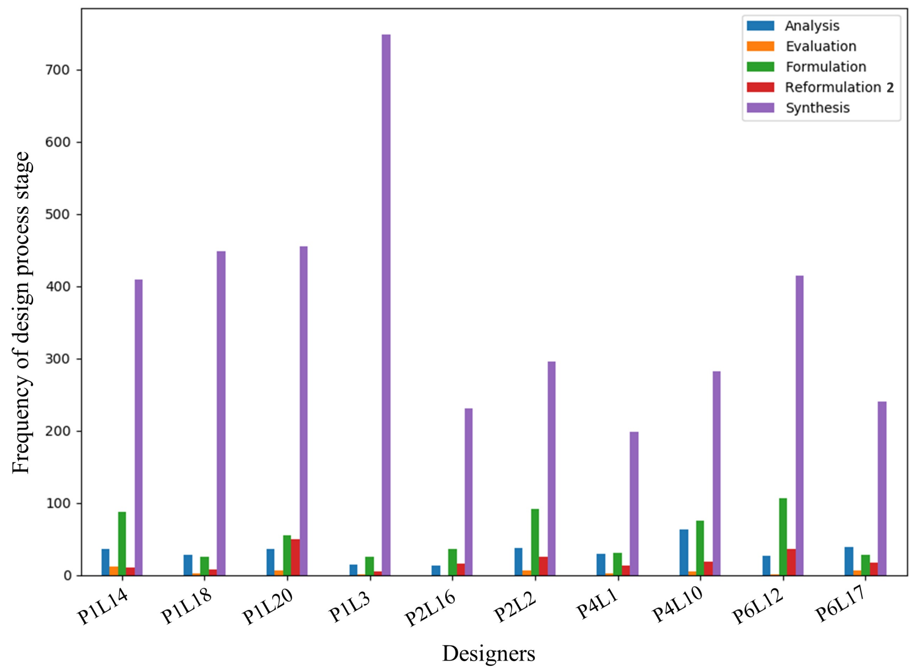


Figure 6.9: Preference of design process stage of cluster 3

involved in editing design components more frequently than the other designers during the design process. Additionally, we observe a higher number of usage of *Formulation* among the designers in Cluster 3 than those in the other clusters. The average number of the *Formulation* used by Cluster 3 is 62, while the average frequencies in Cluster 1 and Cluster 2 are 35 and 40, respectively. Figure 6.9 shows the design process stage preference of Cluster 3.

For the clustering based on designers reflective thinking behavior, designers in each cluster also follow specific design thinking patterns. For example, the designers of Cluster 1 often wonder 1s-3s in between every two actions. This behavior may indicate that the designers in this cluster prefer trial-and-error, thus quickly clicking different design action buttons in the CAD software to explore the design space. In Cluster 2 and Cluster 3, designers follow a similar distribution of time gaps. However, unlike Cluster 1, designers of Clusters 2 and 3 has a relatively lower number of 1-3s time gaps. Rather in these clusters, 4-10s time gaps are prominent. This indicates that the designers in these clusters tend to ponder a little bit before taking the next design action. There is only one designer in Cluster 4. In different from the other students, this student has a uniform distribution of the time gaps during the entire design process.

The clustering of the design embedding obtained from the one-step sequential behavior indicates that designers follow several design patterns. For example, we observed that designers in two clusters use *Synthesis* $\longrightarrow$ *Synthesis* and *Formulation* $\longrightarrow$ *Synthesis* very frequently.

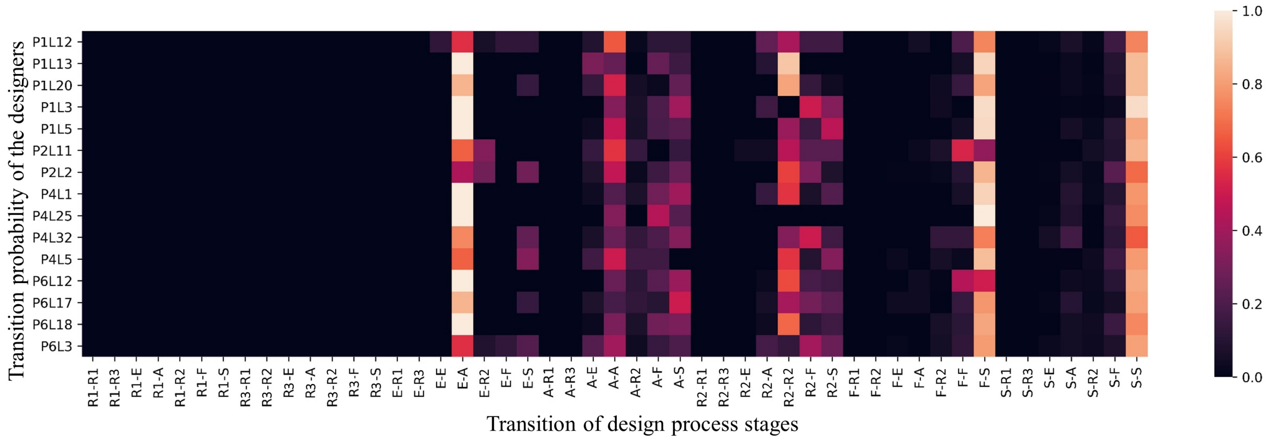


Figure 6.10: Heat map of the transition probability of the design patterns of cluster 2

Synthesis $\longrightarrow$ *Synthesis* action pair indicates that designers sequentially edit the parameters of design components. For example, after changing the solar panel's tilt angle, designers continue changing the azimuth of it. *Formulation* $\longrightarrow$ *Synthesis* action pair indicates that after adding a component, a designer starts to edit its parameters. For example, after adding a solar panel, a designer starts changing the solar panels' base height. There are some design patterns that are distinct from each cluster. For example, the designers in Cluster 2 use *Evaluation* $\longrightarrow$ *Analysis* design patterns a lot during their design processes, while this pattern is used very rarely among the designers in Cluster 1. This pattern indicates that after doing cost evaluation (compare the current cost with the given budget), the designer then analyzes the systems energy output. Figure 6.10 shows a heat map of the transition probability of the design patterns found by the designers of Cluster 2. The bright square indicates a high transition probability of the corresponding design patterns, where the dark square indicates no or very low transition probability.

6.5 Conclusion

In this study, we develop an approach to represent design thinking by characterizing design behaviors from multiple dimensions. We identified five different design behaviors, including design action preference, one-step sequential behavior, contextual behavior, long-term sequential behavior, and reflective design thinking. The design behaviors are characterized by different machine learning and statistical methods, and the design thinking is represented through a latent representation referred to as design embedding. We use the distribution of design actions to characterize designers action preferences. The First-order Markov model is utilized for characterizing designers one-step sequential behavior. To model designers short-term sequential behaviors, the Doc2Vec sequence learning technique is adopted, while a bi-directional LSTM autoencoder is used to characterize the long-term sequential behavior. Finally, we use time gap distribution to represent reflective design thinking. After identifying the design embedding from each design behavior, the X-mean method is applied to cluster each embedding to identify similar behavioral patterns. The result indicates that the behavioral patterns characterized in different dimensions do not necessarily categorize designers in the same cluster. Also, while designers are clustered based on their design behavioral patterns, different design patterns could lead to similar design quality. The major contribution of this paper is the identification of latent representation (i.e., design embedding) of design thinking through design behaviors from multiple dimensions. The implementation of design embedding can be useful in design research in different ways. For example, design embedding can be used to identify designers with similar behavioral patterns and discover beneficial design strategies. Furthermore, as a design process is typically a combination of different design behaviors, different forms of design embeddings can be integrated to develop predictive models that could yield better accuracy for design performance forecasting.

7 Predicting Design Decision Using Deep Learning Method

7.1 Overview

In engineering systems design, designers iteratively go back and forth between different design stages to explore the design space and search for the best design solution that satisfies all design constraints. For complex design problems, human has shown surprising capability in effectively reducing the dimensionality of design space and quickly converging it to a reasonable range for algorithms to step in and continue the search process. Therefore, modeling how human designers make decisions in such a sequential design process can help discover beneficial design patterns, strategies, and heuristics, which are important to the development of new algorithms embedded with human intelligence to augment the computational design. In this chapter, we develop a deep learning-based approach to model and predict designers sequential decisions in the systems design context. The core of this approach is an integration of the function-behavior-structure model for design process characterization and the long short-term memory unit model for deep leaning. This approach is demonstrated in two case studies on solar energy system design, and its prediction accuracy is evaluated benchmarking on several commonly used models of sequential design decisions, such as the Markov Chain model, the Hidden Markov Chain model, and the random sequence generation model. The results indicate that the proposed approach outperforms the other traditional models. This implies that during a system design task, designers are very likely to rely on both short-term and long-term memory of past design decisions in guiding their future decision making in the design process. Our approach can support human-computer interactions in design and is general to be applied in other design contexts as long as the sequential data of design actions are available.

-

7.2 Research Approach and Technical Background

In this section, we first introduce our research approach. Then, we introduce the technical background regarding the deep learning models and the function-behavior-structure (FBS) design process model adopted in our approach.

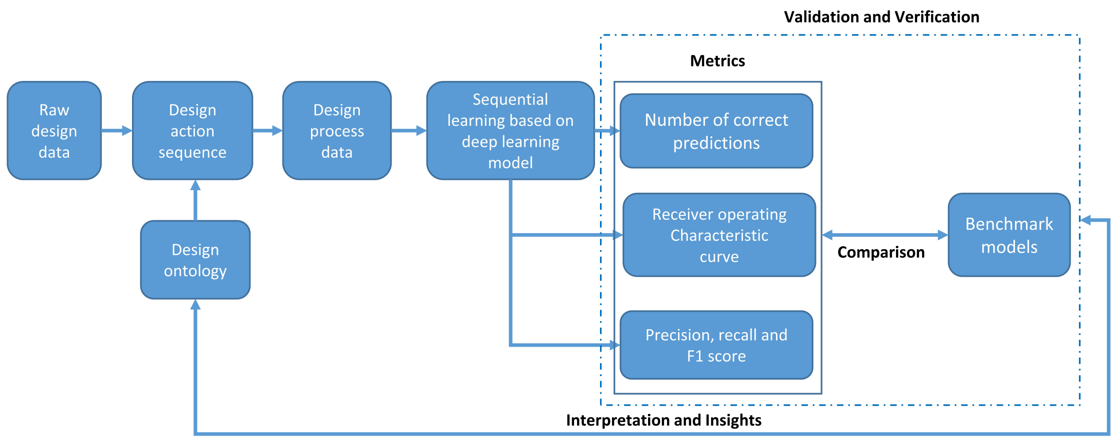


Figure 7.1: The overall research approach for predicting design decisions

7.2.1 The research approach

The approach starts with the raw data collection of designers sequential design decisions from different sources such as the action logger of computer-aided design (CAD) software, interviews of designers, design documents, etc. The raw data contains the details of human design behaviors (i.e., design actions) as well as design artifacts information, such as values of design parameters, simulation results, etc. In this study, we only extract the actions which are only design-related; for example, in a CAD environment, these actions could be adding a new component or editing that component. Designers act based on the given design requirements and constraints; thus, those design actions can reflect designers thinking and strategies in searching the design space. Next, we apply a design process model to convert the design actions into design process data. The design process model consists of a series of design stages that characterize a design process. This treatment transforms the action space into a design process space. This treatment helps better interpret and understand designers sequential design thinking and reduce the dimensionality of the sequential action data (see Section 7.3.2 for details). Then, we use the sequential design process data to train deep learning models and predict the next immediate design action category (i.e., the design stage defined by a design process model) based on the trained models. In this study, we use ANN models, particularly the FNN and RNN models, to implement the deep learning approach as these two models can properly handle sequential data [141]. Finally, we evaluate the predictive performance of these models and compare them with those commonly used models using different metrics, such as testing accuracy, precision, recall, F1 score and area under the receiver operating characteristics (AUROC) curve, at both aggregated level and design process stage level (see Section 7.4 for details). Figure 7.1 depicts a schematic diagram of the overall approach used in this study.

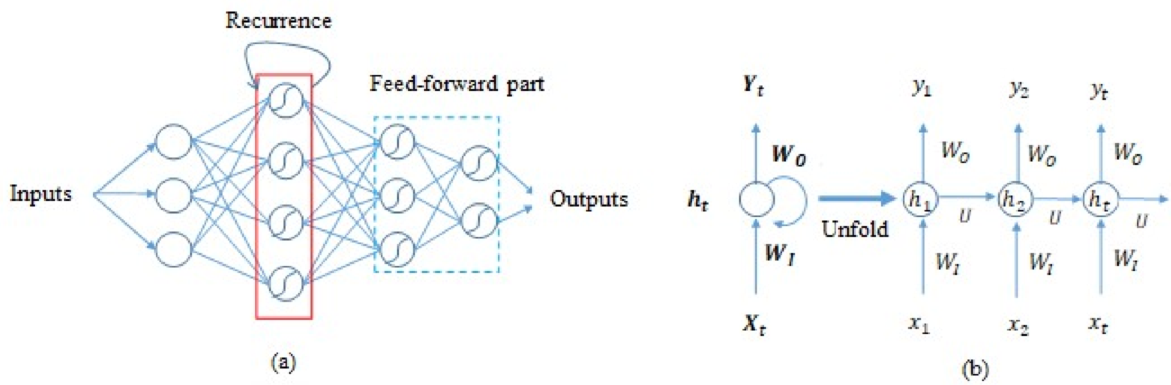


Figure 7.2: (a) Standard recurrent neural network architecture with feedback loop; (b) Unfolded recurrent neural network

7.2.2 Feed-forward neural network and recurrent neural network

In the feed-forward neural network (FNN) architecture, the information follows one direction from input to output with no back loops. In addition to the input layer and output layer, FNN may have single or multiple hidden layers. When FNN has only one layer between input and output, it is known as the single-layer perceptron. An FNN with more than one hidden layer, including the output layer, is called a multilayer perceptron. An FNN with a single hidden layer between the input layer and the output layer is often sufficient to be universal function approximation [142]. However, deep neural networks with additional hidden layers outperform this shallow model. A standard depth of deep neural network (i.e., the number of hidden layers) may vary from two or three to even one thousand [143]. The structure of the recurrent neural network (RNN) is similar to that of an FNN. The only distinction is that there is no restriction on back loops. So, the information not only passes in one direction forward but also does it flow backward, called recurrence (see Figure 7.2). This feature allows RNN to create a hidden state which carries information from the previous time steps to its current step. Although RNN can be used for capturing long-term dependencies, simple recurrent units are not effective in this task due to vanishing gradient problem [32]. To solve this problem, Hocreiter et al. [139] proposed the long short term memory units (LSTM), a special type of mechanism where information flow is controlled by three different gates, namely input gate, forget gate and output gate. LSTM is more widely used than simple RNN in many domains for its capability of modeling long-term dependencies. To study to what extent the past decisions of designers can influence their future decision-making, we adopt LSTM as a representative RNN model in our study.

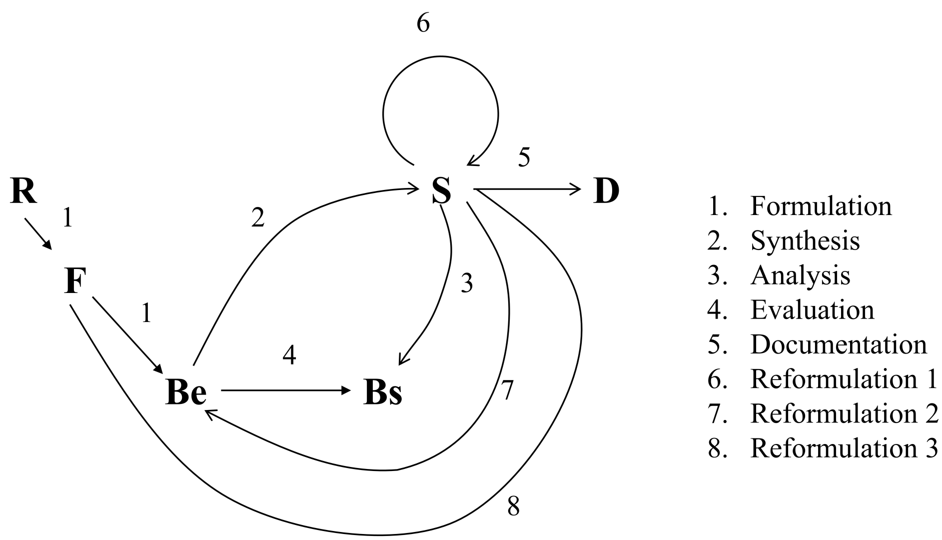


Figure 7.3: The Function-Behavior-Structure (FBS) design ontology (adapted from [1])

7.2.3 The Function-Behavior-Structure design process model

The Function-Behavior-Structure (FBS) model [1], a domain-independent design process model, consists of three ontological design variables: Function (F), Behavior (B), and Structure (S). Function describes the purpose of the design and establishes the connection between design goals and measurable effect. Behavior is defined as a design attribute that can be derived from the design structure. Structure (S) is defined as the design component and its interconnected relationship. During the design task, designers establish interconnections among these three variables. The first basic interconnection is constructed by transforming the function into behavior and behavior into the structure, i.e., $F \longrightarrow B$ (#1), and $B \longrightarrow S$ (#2), as shown in Figure 7.3. Here behavior is interpreted as the expected performance in order to achieve the function. However, once the structure is generated, the expected performance may not be achieved. Therefore, the performance from the structure needs to be compared with the expected performance. For this reason, in the FBS design model, the behavior is distinguished into two separate classes of behavior: expected behavior (Be) and behavior derived from the structure (Bs). With these additional variables, the transformation is extended as follows: $F \longrightarrow Be$ (#1), $Be \longrightarrow S$ (#2), $S \longrightarrow Bs$ (#3), and $Be \longrightarrow Bs$ (#4), as shown in Figure 7.3.

Often designers start their design from the initial requirement and finish the design task by reporting the description of the design. Therefore, two additional design variables: requirements (R) and description (D) are added to the design process model. The FBS design process model regards requirements (R) as a function (F) generator and defines description (D) as the representation of a design task. The additional transformations are $S \longrightarrow D$ (#5) and $R \longrightarrow F$ (#1)

Table 7.1: Transformation of ontological design variables and the rationale of the FBS-based design processes

Transfor-mation	Design Process	Definition and Interpretation
$F \longrightarrow Be$ & $R \longrightarrow F$	Formulation	Generate when requirement is transformed into function and function is transformed into behavior.
$Be \longrightarrow S$	Analysis	Obtain behavior from generated structure
$S \longrightarrow Bs$	Synthesis	Generate and tune structure based on the expected behavior.
$Be \longrightarrow Bs$	Evaluation	Comparison of the expected behavior and actual behavior
$S \longrightarrow D$	Documentation	Generate design description based on the structure
$S \longrightarrow S$	Reformulation 1	Regenerate and modify one structure to another structure
$S \longrightarrow Be$	Reformulation 2	Regenerate or modify structure based on the expected behavior
$S \longrightarrow Be$	Reformulation 3	Regenerate and modify structure based on the formulation

During the design task, designers iteratively and incrementally improve the design. Also, designers implement new ideas by removing or changing the existing structures or functions in order to improve the behavior of the structure. Thus, additional three transformations are included in the FBS model: $S \longrightarrow S$ (#6), $S \longrightarrow Be$ (#7), and $S \longrightarrow F$ (#8). With all of these transformations, a total of eight design processes are obtained, as summarized in Table 7.1, along with the interpretations. This FBS design ontology provides us with the rationale for the transformation of design action data to design process data in support of the study of design thinking.

7.3 Predicting Sequential Design Process in Solar Energy Systems Design with Two Case Studies

In this section, we present two case studies on solar energy systems design and implement the proposed approach to predicting designers sequential design decisions. First, we introduce the design experiments conducted for data collection. Next, we present the collected data and introduce the methods for processing it.

7.3.1 The design context

In order to collect sequential design behavioral data, we conducted a series of design challenges on real-world engineering design problems. The challenges were held at the University of Arkansas. Both undergraduate and graduate students from engineering disciplines participated in

Table 7.2: Design requirements of the two design Problems

Design challenges	Design variables	Design constraints
Energy-plus home design	Story	1
	Number of windows	>4
	Size of windows	>1.44 m^2
	Number of doors	1
	Size of doors ($Width \times Height$)	>1.2$m \times 2m$
	Height of wall	>2.5 m
	Distance between ridge to panel	>0
Solarized parking lot design	Base height	$\geq$ 3.5 m
	Tilt angle	$\leq$ 20
	Solar panel rack	Shall not produce any hindrance to the pedestrian zone and drive ways
	The pole of rack	Shall be placed along with the parking lot line marker

these challenges. In this study, we mainly adopt the data collected from two design challenges. In the first challenge, the students were asked to design a solarized home in Texas with a budget of $200,000 (see Figure 7.4). The second challenge was to build a solarized parking lot at the University of Arkansas. The budget for this challenge was $1.5M.

The design objective of both challenges is to maximize the annual net energy (ANE) with the given budget. The design requirements and constraints are provided to the participants so they can start the design with a focus on configuration and parametric design. In this study, the design variables are mainly related to the components that have a direct impact on the design objective. We limit the number of design variables and design constraints so that the experiment remains in a controlled manner. By doing so, we are able to compare the models in different settings of design complexity. Table 7.2 shows the requirements and constraints of both design challenges. These two design problems reflect different design complexity in terms of design variables and the couplings between these variables. For example, the energy plus home design problem has more design variables than the parking lot design problem. Therefore, the two design problems bring

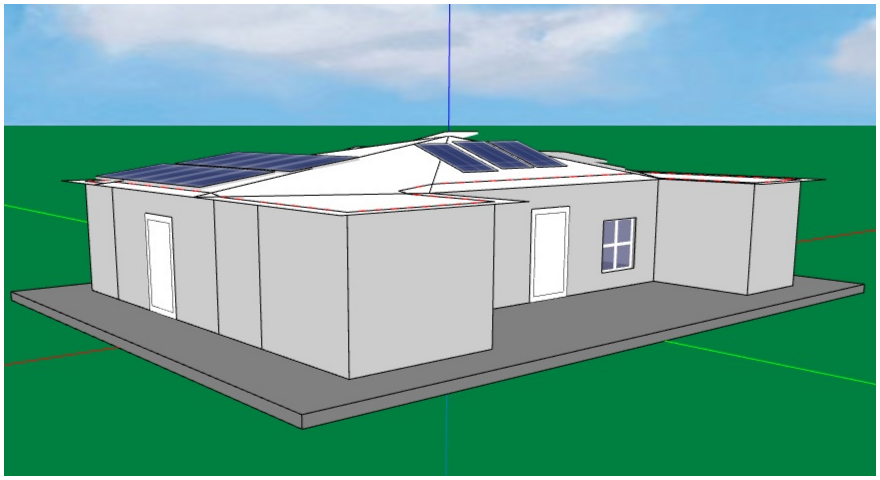


Figure 7.4: An example of solarized home design problem designed by one of the participants

generality for a comparison of the proposed approach.

Students designs were conducted within a computer-aided design (CAD) environment, called Energy3D. Energy3D is a full-fledged CAD software specially built for solar systems design [35]. It has several unique features, such as interactive visualization, high-fidelity simulation, and built-in financial evaluation. These features can help designers effectively explore and exploit the design space. Moreover, Energy3D has a non-intrusive data action logger. That means designers are not aware of the data collection process, and this helps reduces participants cognitive burden that could be introduced in experimental settings. As a result, the data that reflects designers thinking and decision-making can be less biased. Energy3D sorts and logs every performed action in JSON format. This high-resolution data provides us with a large amount of data that is essential to implementing deep learning models.

7.3.2 Data collection and preprocessing

Energy3D collects the continuous flow of design action data, which includes time-steps, design actions, design parameter values, and simulation results. In the solarized home design problem, a total of 52 engineering students participated in this design challenge. Among them, 29 students are undergraduate students, and 23 are graduate students. 40 students are from Mechanical Engineering. On average, the design action log records 1500 lines and 220 intermediate files per student. In the parking lot design problem, a total of 41 students participated in which 35 students are from undergraduate, and 5 students are from graduate students, all in the major of Mechanical Engineering. The design action log records, on average, 1500 lines of data per student.

Table 7.3: Mapping of design actions to design process stage

Design process	**Design actions**
Formulation	Add any component
Analysis	Analysis of annual net energy
Synthesis	Edit any component
Evaluation	Cost analysis
Reformulation 1	Remove structure
Reformulation 2	Remove solar device
Reformulation 3	Remove other components

We ignored the actions, such as camera, add tree that do not have direct effects on the design outcomes (i.e., ANE). After removing those irrelevant actions, there are about 300 actions per participant on average, and 115 are unique actions in the solarized home design problem. In the solarized parking lot design problem, the average number of design actions is about 350 after removing those trivial actions. Among these, 72 design actions are unique.

Analysis of such a high dimension action space would yield results hard to interpret. To better understand the design process and designers sequential decision-making strategies, the FBS-based design process model introduced in Section 7.2.3 is applied. In this study, an encoding scheme (see Table 7.3) is established to transcribe different types of design actions to the seven design process stages including Formulation (F), Analysis (A), Evaluation (E), Synthesis (S), Reformulation 1 (R1), Reformulation 2 (R2), and Reformulation 3 (R3). In our design problem, adding any component such as add wall, add a solar panel, etc. refers to Formulation. Designers add components in order to construct the artifact to achieve the desired objective. According to Table 7.1, Synthesis occurs when parameters of a component are tuned to achieve the expected behaviors. So, the action of editing any components refers to Synthesis. When designers analyze the ANE of their solar system designs, this action refers to Analysis because designers aim to obtain the behavior from the generated structure. To compare the expected behavior and the actual structural behavior, designers check whether the design cost exceeds the given budget or not. According to Table 7.1, this can be defined as Evaluation. Finally, designers remove structures and regenerate new structures to meet the design requirements and their own intrinsic criteria. When designers remove structure related components such as walls, windows, or doors, these actions are referred to as Reformulation 1. However, when designers remove the roof, this action is primarily driven by the obtained expected behavior of design, e.g., the ANE does not meet the objective. So, they modify

Design process stages	F	A	E	S	R1	R2	R3
Formulation	1	0	0	0	0	0	0
Analysis	0	1	0	0	0	0	0
Reformulation 1	0	0	0	0	1	0	0
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
Analysis	0	1	0	0	0	0	0

Figure 7.5: One-hot vector representation of a sequence

the roof style in order to put more solar panels for the potential increase of ANE. According to Table 7.1, these actions can be defined as Reformulation 2. Finally, if designers remove other structures, such as trees, these design actions are defined as Reformulation 3 because these modifications are merely based on the formulation process. We did not consider Documentation because Energy3D automatically documents all the design process. Therefore, designers are not required to report their designs separately. The application of the FBS model helps map the design action space to the design thinking space to better understand the design rationale and the discovery of sequential design patterns. This treatment also helps dimension reduction that is useful to reduce the effect of the curse of dimensionality [144]. Given a set of sequential text data, we must encode the sequences so that it can be implemented by neural networks. The most popular encoding technique is known as one-hot encoding [145]. One hot encoding transforms a single variable of n observations with m distinct variables into m binary variable with n observations. Each observation indicates the presence (1) for the corresponding position of that variable and absence (0) in all other dimensions. Figure 7.5 shows an example of the one-hot vector presentation of a design sequence.

7.4 Result and Discussion

In this section, we present the results of the LSTM and FNN models and compare them with the models that are commonly used in existing literature, such as the Markov model (MM) and the hidden Markov model (HMM). Additionally, we develop a repetitive model (REP model) for comparison because we found from our previous section 5 that designers quite frequently repeat the previous design action in the CAD environment. For example, we found that in the solarized home design problem, on average, 51.2% of design actions were simply repeating the action in the previous step. In the solarized parking lot design, we found the repetition rate is at about 59.3%. So, in the REP model, we simply use the average percentage of occurrence of each design stage as the model to predict the next design stage with the highest percentage value. Finally, a random model is presented as the benchmark for all the models in comparison. The purpose is to examine whether the design sequences indeed follow certain patterns or just randomness. In this study, since there are seven design process stages, the prediction of the next stage will be a random selection of one process stage from seven following a uniform distribution. Thus, every process stage has a probability of 1/7 to be selected.

In the following two sections, we first evaluate the performance of different models in terms of prediction accuracy, precision, recall, and F1 score regardless of the category of design actions (i.e., the design process stages defined by the FBS model). Next, we perform an in-depth analysis of how accurately each design process stage in the next step can be predicted and compare the performance of different models using the metrics of the area under the receiver operating curve (AUROC).

7.4.1 Evaluation of model performance at the level of the entire sequence

To validate the models, we adopt the k-fold cross-validation [146] technique, where we divide our data into five folds. First, we use any four folds to train the models and leave the remaining fold for validation purposes. Next, we train the models on a new combination of 4 folds, including the previously withheld fold, and validate the model again with the remaining one. In this way, we iterate through all over the five rounds. An illustration of the 5-fold cross-validation method is shown in Figure 7.6.

Keras deep learning library [147] is used to run the HMM, FNN, and LSTM model, and we programmed for the Markov chain model. While going through each of the rounds, the training data set performs forward pass and backward pass (a.k.a. backpropagation) [148] in order to update

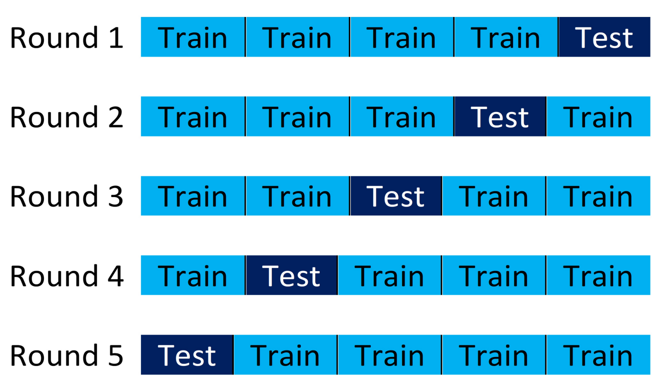


Figure 7.6: Training and testing data split according to 5-fold cross validation technique

the models parameters (including both weight values and bias values).When the entire dataset is passed forward and backward through the neural network, its called one epoch. During testing, we predict the next action ($a_{(t+1)}$) by passing the previous actions from time 0 to t as the input into the trained model. So, if a design sequence has n actions, then n-1 predictions will be made. Then, by comparing with the real observation of a design sequence, we count the total number of correctly predicted actions (n_{cp}) and divide it by the total number of predictions, i.e. n-1. In this way, we get the prediction accuracy of that model in every epoch. In this study, only the prediction accuracy of the last epoch (when the model is fully trained) in each round is taken, and the average from five rounds is used as the metric for evaluating a models predictive power. The mathematical expression of this metric is as follows:

$$Predictionaccuracy = \frac{1}{R}\sum_{i=1}^{R}\left(\frac{n_{cp}}{n_i^{max}-1}\right) \tag{7.1}$$

where R denotes the number of rounds for cross-validation. R=5 in this study. n_i^max is the maximal number of actions of the design sequence (i.e., the length of the longest design sequence) in round i. The models were trained by stochastic gradient descent [149] algorithm with a learning rate of 0.001. This learning rate is determined by trial and error for producing the best accuracy, and we found all the models converge after 40 epochs. Section 7.4.3 presents a sensitivity analysis of the hyperparameters.

MM provides the prediction of the design process state in the next step based on the state in the current time step. With the design sequence as input, training of an MM will produce a 7×7

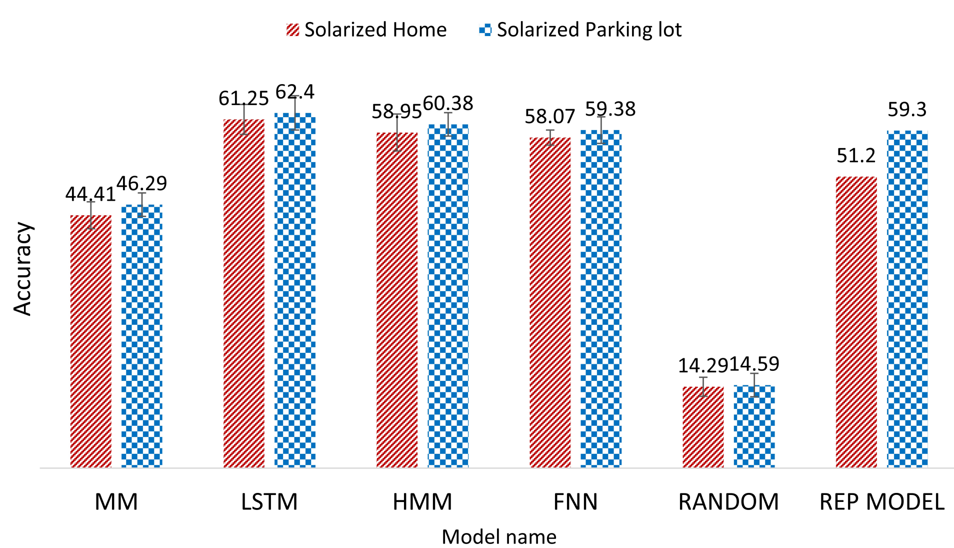


Figure 7.7: Testing accuracy of different predictive models

transition probability matrix because the seven design process stages in the FBS model represent seven states in MM. Each of the entries of the matrix defines the probability of one design stage transitioning to the next design stage. We follow the same structure in Figure 7.6 to train and test MM. The trained model (i.e., the transition probability matrix) is an aggregation by averaging the matrices obtained from every designer in the training dataset. When testing a MM model, for every given FBS design process stage in a design sequence, the MM will predict seven probabilities of design stages following that given stage, and the one with the highest probability is picked for comparing with the real data, and then the prediction accuracy is reported. MM is not associated with any external parameters. So, we calculate the prediction accuracy just based on the transition probability matrix.

Figure 7.7 shows a comparison of the testing accuracy of all the models for both the solarized home design and the solarized parking lot design. The baseline model, i.e., the random model, shows the least accuracy of 14.29% (standard deviation 1.66%) and 14.59% (standard deviation 2.02%) for solarized home and solarized parking lot, respectively. The accuracy of the REP model is much higher than the random model, which is about 51.2% for the solarized home dataset and 59.3% for the solarized parking lot dataset. This is because the REP model is developed based on the simple repetition process in design, and it is observed that when designers were working on this design problem using CAD software, many of them repeated their previous action very frequently, and the REP model captures such a pattern. We present only the prediction accuracy for the random model and the REP model because the training process is not needed in these models. The accuracy of the REP model for the solarized parking lot design is higher than the solarized home design. This is probably due to the reason that the number of unique design actions in the solarized parking lot design is lower than those in the other design problem. Therefore, the design actions can be more frequently repeated in the former design context, and this is well captured by the REP model.

It is shown from Figure 7.7 that both MM and HMM yields better performance than the random model for both datasets. In the solarized home design problem, the prediction accuracy of MM and HMM are 44.41% and 58.95%, respectively. Similarly, in the solarized parking lot, the accuracies for MM and HMM are 46.29% and 60.38%, respectively. This indicates that designers actions indeed follow certain patterns and are not random. In MM, each action is dependent only on the action of the previous step. As a consequence, MM does not encode long-term memory of past events in the prediction. On the other hand, the inclusion of hidden-state architecture in HMM allows it to remember a few past state. As in design, designers do have to refer to past information in guiding their future design decisions, the successful modeling of past information into the hidden state may be the reason why HMM has significantly higher prediction accuracy (the average is 58.1%) than those of MM and REP model. This observation echoes many of the existing studies on the comparison between MM and HMM, and the conclusion that HMM outperforms MM [41] is very likely due to the reason that HMM can better model the interdependencies between past states and current state.

We also observe that HMM slightly outperforms the FNN model on average. FNN gives an average prediction accuracy of 58.07% (with a standard deviation of 2.14%), which is 0.88% lower than that of the HMM in the solarized home design. For the parking lot dataset, FNN achieves a prediction accuracy of 59.38% (with a standard deviation of 2.3%), which is 1% lower than HMM. The ability to pass information from the previous states to the current hidden state makes HMM a better predictor than FNN in this study. FNN does not essentially have a hidden state as it does not consider feedback loop in hidden layers and hidden units are not connected (see Section 7.2.2 for the architecture of FNN).

Among all of the models, LSTM produces the highest prediction accuracy in both datasets. The prediction accuracy for the solarized home design and the parking lot design is 61.25% and 62.4%, respectively. The significance of the difference between the deep learning models and the existing models are assessed using the paired t-test. Among the existing models, the prediction accuracy of the HMM is close to the deep learning models (i.e., LSTM and FNN). Therefore, using this instance, the null hypothesis (H_0) of the test is that the mean of the prediction accuracy of the deep learning models is equal to that the HMM model. The alternative hypothesis (H_a) is that the mean of the accuracy of the deep learning models is higher than that of the HMM. With the level of significance 0.05, the p-value indicates (0.04756 for the solarized home design and 0.03234 for the solarized parking lot design) that, for both datasets, the prediction accuracy of the LSTM model is significantly higher than that of the HMM; however, HMM is not significantly better than FNN as indicated by the p-values (0.2339 and 0.2088 for solarized home and solarized parking lot, respectively). The results imply that during the systems design process, designers future actions do have strong dependencies with the past design information. For example, in the solarized home design problem, we observe that one designer, most of the time, analyzes Building cost after adding several new components, such as $Addwindow \longrightarrow Editwindow \longrightarrow Buildingcost$, and again, $AddRack \longrightarrow EditRack \longrightarrow AddSolarPanel \longrightarrow Buildingcost$. This infers that after adding new components, this designer started configuring the related components to try improving the design performance and check if the total cost is still within the budget. Since LSTM leverages longer memory of past events and their interconnections in predicting future states, its architecture best resembles designers decision-making process, and this is probably the reason why it yields the best performance in this study. LSTMs highest prediction accuracy also implies that designer doesnt only recall short-term memory (like what MM and HMM do), but also use long-term memory information in their design process.

Generally, the models that encode longer memory in their architecture performs better in predicting designers future actions. The possible reason is that in systems design, there are many design variables that are interdependent, and designers may not be able to immediately understand such a complex relationship. One method that can help understand the inter-dependencies among design variables and their effects on the design objective is to constantly change the variables and then run simulations to see how it would affect the objective value. Since there are multiple variables in these design case studies (as presented in Table 1), designers often perform a series of configurations (such as $Addwall \longrightarrow Editwall \longrightarrow EditFoundation \longrightarrow AddRoof \longrightarrow AddRack$ for home design problem and $Addsolarpanel \longrightarrow Editpanel \longrightarrow Changepoleheight$ for solarized parking lot design problem), and then perform ANE analysis cost evaluation. These processes may have to be repeated several times in order to find the best configuration/combination of design variables for the desired objective. Such a pattern can reflect designers exploration-exploitation strategies for design trade-off (i.e., the sequential decision-making strategy) and their design heuristics. The results indicate that the hidden states of LSTM and HMM work as a memory unit seem to well capture those design patterns.

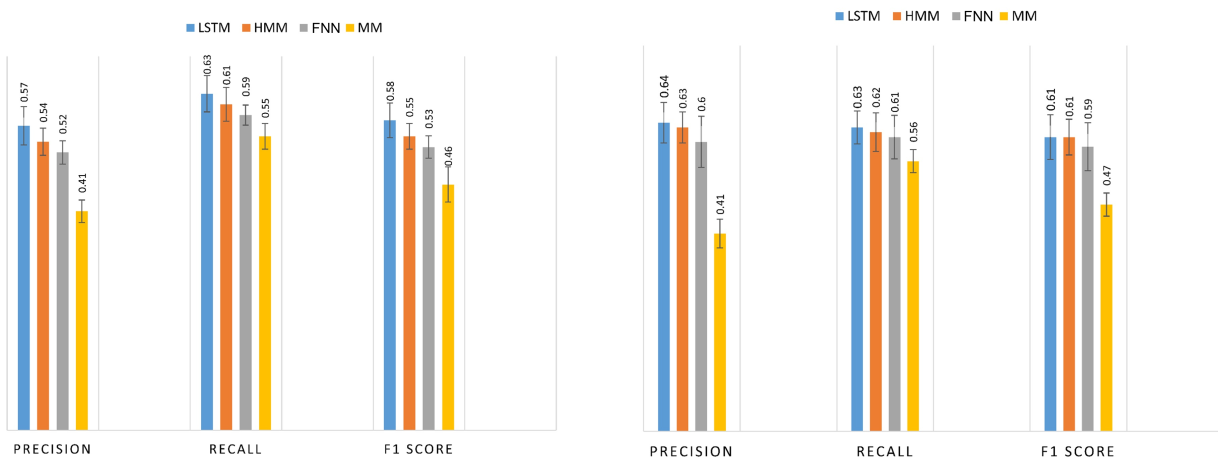
In addition to the prediction accuracy defined in Equation 7.1, we also report the metrics, including Precision, Recall, and F1 score (see Equations 7.2, 7.3 and 7.4) for evaluation.

$$Precision = \frac{Truepositive(TP)}{Truepositive(TP) + Falsepositive(FP)} \tag{7.2}$$

$$Recall = \frac{Truepositive(TP)}{Truepositive(TP) + Falsenegetive(FN)} \tag{7.3}$$

$$F1score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{7.4}$$

These three metrics are often used simultaneously, as each of these metrics reveals different aspects of a models predictive power [150]. For example, there might be a case where precision is higher, but the recall is lower than the other models. In that case, for a proper evaluation, the F1 score, which takes the harmonic means of precision and recall can be used. Figure 7.8 shows the result of the scores of the metrics of different models for both case studies. Among all the models, LSTM outperforms the other models, especially in the solarized home dataset. LSTM archives about 0.57, 0.63, and 0.58 for precision, recall, and F1 score respectively while the nearest scores for HMM are 0.54, 0.61, and 0.55, respectively. The same conclusion holds in the parking lot dataset.



(a) Solarized home design

(b) Solarized parking lot design

Figure 7.8: F1, Precision and Recall score for different models

7.4.2 Evaluation of model performance at each category of design actions

In order to understand the models performance at a finer resolution, we check how well each model can predict each category of design actions, i.e., the design process stage defined by the FBS design process model. To achieve this, we adopt receiver operating characterizes (ROC) curve [151] as the method evaluates the models two operating characteristics (the true positive rate and the false positive rate) on each design process stage under different binary threshold values from 0 to 1.

After obtaining the ROC curves for each design process stage, the area under the ROC curve (AUROC) is used to provide one single metric which aggregates the predictive performance cross the thresholds so that we can compare in which design process stage does the model perform better. A larger AUROC indicates a better predictive performance. Figure 7.9 shows an example of LSTMs ROC curve of each design process stage in one fold of prediction for the solarized home design problem. If the AUROC from all the five folds are averaged, we obtain Figure 7.10. For example, the AUROC of Formulation and Analysis for the LSTM model in the solarized home design problem reaches a maximum of 0.82 and 0.80, respectively, among the seven design process categories. LSTM model also produces a good AUROC score (0.77) for Evaluation. These results imply that designers tend to enter into these design stages after completing a certain series of design tasks. For example, the designers must first construct the house, which involves many design actions related to Formulation (i.e., Add Wall, Add Window, etc.) and then evaluate the performance by simulating annual net energy (Analysis) and analyzing the building cost (Evaluation). However, in the solarized parking lot problem, Evaluation achieves the highest AUROC score (0.91), which indicates that designers follow certain strong behavioral patterns while checking the design cost. Also, the AUROC scores for Formulation (0.83) and Reformulation 2 (0.85) are higher than the other design process stages. The potential explanation is that as solarized parking lot design is less complex, the only design component that designers add and remove is the solar panel. While designers are adding solar panels, they frequently check if the overall design cost (i.e., Evaluation) exceeds the budget limit or not. Also, in order to increase the ANE, designers may need to remove existing solar panels (i.e., Reformulation 2) and try a new layout. These patterns of design behaviors make Formulation and Reformulation 2 more easily to be captured by the model.

LSTM also produces some lower AUROC values, particularly for Reformulation 1 and 3, in both datasets. This is because Reformulation involves the design actions of removing components, such as remove a tree or remove a window. These removal actions are often paired with another Reformulation and/or Formulation actions, such as add a wall or add a window. These action pairs reflect designers fine-tuning behaviors (exploitation) on particular design components immediately based on the observations from the CAD interface and no necessary to run a simulation for feedback

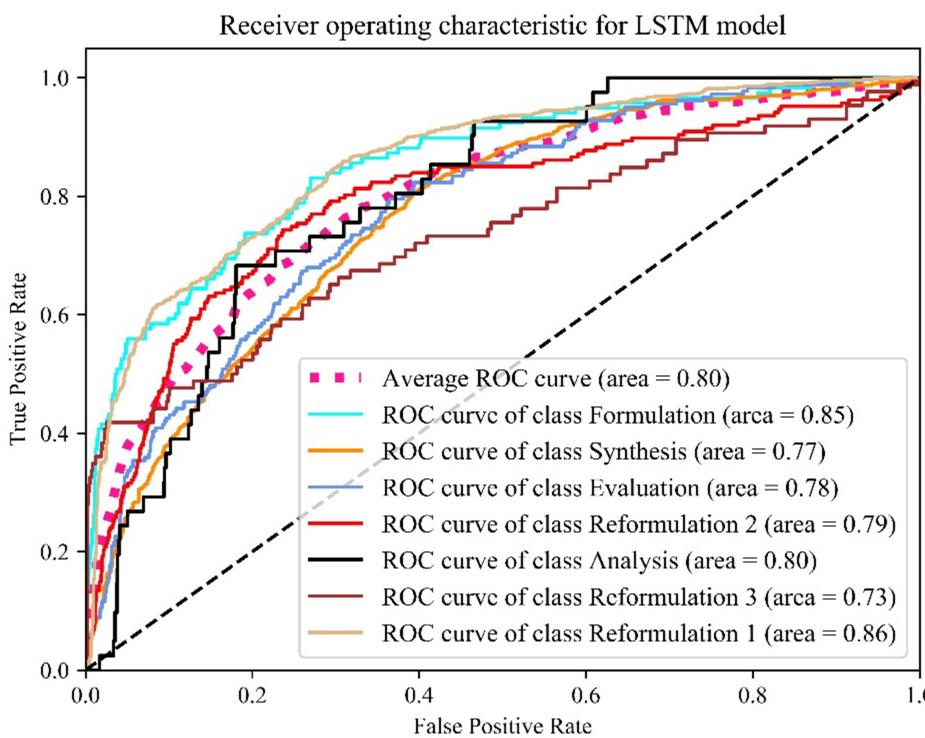
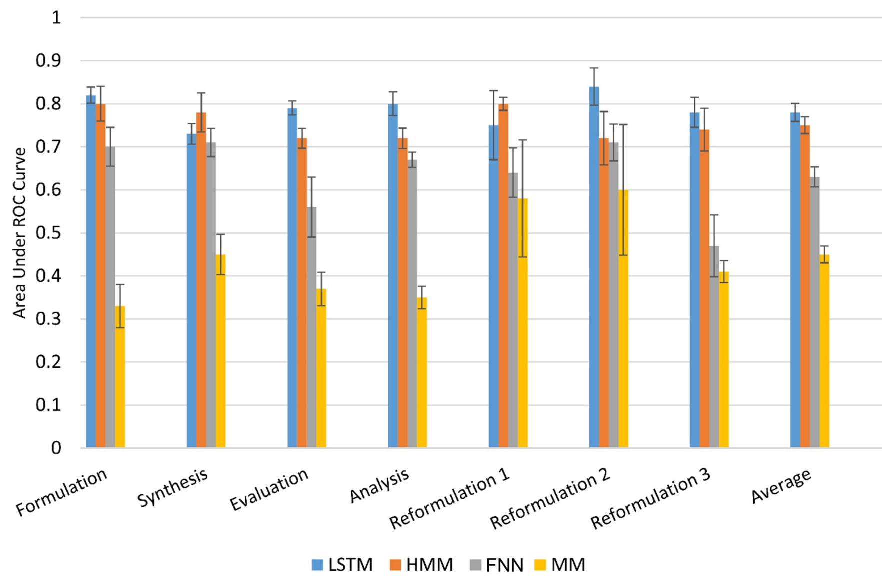
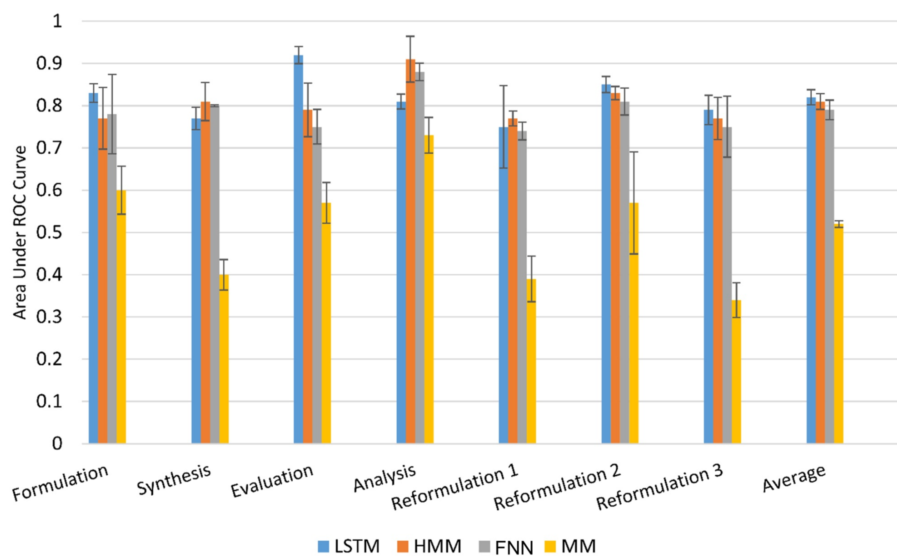


Figure 7.9: Receiver operating characteristics (ROC) curves of LSTM in fold 5 for solarized home design dataset. Each curve represents the model performance for each design process stage

to support their design decisions. Therefore, referring to the action in the last step should be sufficient for prediction, and it does not require to use long-term memory in predicting these design stages. This may also be the reason why MM can produce higher AUROC scores for Reformulation 2 (0.60 in the solarized home design and 0.57 in the parking lot design, respectively). For example, Reformulation 2 contains the design actions related to the removal of solar panels. As solar panels



(a) Solarized home design



(b) Solarized parking lot design

Figure 7.10: Area under the receiver operating characteristics curve (AUROC) score for different models. The Average in last column is the average AUROC value of all design process stages per model

directly affect the system performance, most designers spent a significant amount of time fine-tuning (e.g., add, remove and then add back again) this component, and therefore, there exist a large number of action pairs of Add Solar Panel Remove Solar Panel in the design sequence. Since MM predicts the state only one-time step ahead based on the current state, it captures this design pattern very well. But, on the other hand, it does not effectively capture the patterns that involve longer historical information, such as, Evaluation and Analysis stages in the solarized home design problem, as compared to the other models. However, in the parking lot dataset, the more frequent short-term action pairs are observed, such as *Evolution* $\longrightarrow$ *Evaluation* and *Analysis* $\longrightarrow$ *Evaluation* (see Table 7.4 for the transition probabilities from other design processes to Evaluation as an example), this pattern can be better captured by MM; therefore their AUROC scores are higher.

Table 7.4: Transition probability of Evaluation to other design process stages in both datasets

	Solarized home design	**Solarized parking lot design**
Transit to / Transit from	Evaluation	
Analysis	0.197	0.23
Evaluation	0.061	0.2
Formulation	0.036	0.012
Reformulation 1	0.034	0.002
Reformulation 2	0.036	0.07
Reformulation 3	0.051	0.021
Synthesis	0.08	0.31

Please note that in the solarized house design problem, MM has the least AUROC for Formulation. This is because MM is derived from the frequency of the event. In the design, the repetition of Formulation (corresponds to adding components) does not occur frequently. For example, once a designer finishes adding all the necessary components, e.g., Add Wall and Add Window, she/he would never take those actions again because the house has already been established. Instead, she/he tends to start fine-tuning the associated parameters through the actions of Edit Wall and Edit Window (i.e., the auctions related to Synthesis).

If we take an average for the AUROC scores from every design process stage, that average value can be used to compare the performance between different models, as shown in the last column of Figure 7.10. For the solarized home design problem, we observe that, on average, the LSTM model outperforms the other models with the AUROC of 0.78. The HMM model (0.75) and FFN (0.63) achieve a lower AUROC score than the LSTM model. This indicates that even if HMM and FNN take historical design information into their prediction, they do not effectively process that information during the model training as what LSTM does. Both FNN and HMM perform relatively better on average across all the design process stages than the MM (0.45).

In the parking lot design problem, the LSTM outperforms the other models as well (0.82). However, other predictive models such as HMM and FNN also perform better with the AUROC of 0.81 and 0.79, respectively. The AUROC score of each design process stage predicted by the models is also close to each other. This phenomenon indicates that for the less complex design problem where designers use less complex design patterns and fewer types of design actions (i.e., design variables), there are no significant differences between LSTM and other models. However, for complex design problems where various types of design actions exist, there could exist different design approaches to reaching the objective. As a result, designers behavior may follow different patterns that are hard to be captured by models with simple structures, such as HMM and FNN. But LSTMs gate mechanism (e.g., input gate, forget gate, and output gates) seems well to capture and process the dependent relations between different design stages during a design process, therefore, yields the best performance regardless of the complexity of the dataset. These results cross-validate the conclusion we reached from the prediction accuracy results shown in Figure 7.7.

7.4.3 Sensitivity analysis

When training an LSTM model, there are several pre-determined hyperparameters, such as the number of LSTM layers, LSTM size, the number of dense layers, the size of a dense layer, learning rate and dropout value. LSTM size refers to LSTM nodes in each LSTM layer. The fully connected layer indicates the number of layers of the feed-forward network in Figure 7.2. The size of a fully connected layer indicates the number of nodes in each dense layer. Dropout is the value of dropout regularization. In order to prevent the model from overfitting, we use dropout regularization [152] with two different values. The learning rate is the converge rate used in the stochastic gradient descent algorithm in backpropagation.

To investigate how the prediction accuracy would be affected by these hyperparameters, we perform a sensitivity analysis by changing the values of these parameters and study the corresponding prediction accuracies. In the experiment, we use one layer of LSTM for all the settings with a various number of LSTM nodes. Table 7.5 shows the training accuracy and test accuracy of the LSTM models with different hyperparameter settings. From all the settings, it is observed that the model with one fully connected layer performs better (i.e. above 58%) than the models with two fully connected layers (i.e., 56.17% for the solarized parking lot dataset and 54.95% for the solarized home design dataset). Given the same number of fully connected layers and the same fully connected size, a learning rate of 0.1 produces relatively lower performance (57.50%) than those of other settings.

Table 7.5: Different hyperparameter settings for LSTM model

No.	**LSTM size**	**Fully connected layer**	**Fully connected layer size**	**Dropout**	**Learning rate**	**Testing accuracy**	
						Solarized home	Solarized parking lot
1	256	1	7	0.3	0.1	57.50%	56.27%
2	256	1	7	0.3	0.001	61.25 %	62.4 %
3	256	1	7	0.2	0.01	58.97%	60.81%
4	128	1	7	0.3	0.01	59.16%	59.38%
5	128	1	7	0.3	0.1	58.08%	57.49%
6	256	2	128 and 7	0.2	0.1	54.95%	56.17%

But the dropout rate (changing from 0.3 to 0.2) and the LSTM size (changing from 256 to 128 nodes) do not influence the model significantly. Among all settings, it is found that the model with LSTM unit 256, dropout value with 0.3, and learning rate with 0.001 provides the best accuracy in both datasets.

7.4.4 Conclusion

In this study, a deep learning approach is developed to analyze and predict sequential design decisions in the context of systems design. We use Energy3D as the research platform to conduct design challenges and collect designers sequential design behavioral data. Then, the FBS-based design process model is adopted to transform the sequential design action data into the sequential design process data. Based on the design process data, we adopted two deep learning models, i.e., the FNN and the LSTM, to predict designers next immediate design process stage. These deep learning models are evaluated with different performance metrics, including accuracy, precision, recall, F1 score, and area under the ROC curve. Their predictive performances are compared with the other four models, including a MM, an HMM, a repetitive model, and a random model. Predictive power is assessed at the level of the entire design sequence, as well as at the level of each design process stage. We found that, on average, the LSTM model outperforms all the other models, while FNN shows lower performance than traditionally used HMM. From the ROC curve analysis, we found that in both datasets, LSTM yields overall better performance for all of the design process. In contrast, the predictive performance of the other models is not consistent. Moreover, from this study, we also observe that for design problems that are less complex and involve a fewer number of design variables, predictive models perform similarly. For complex design problems, the performance of the predictive models differ. However, regardless of the design complexity, LSTM performs better than the other models. With these findings, we conclude that both short-term and long-term memories have together influenced human sequential design decision-making. The neglect of either aspect in the modeling would lead to inadequate prediction accuracy. However, such an effect is not always significant for all design actions in every stage because, indeed, it is found that in predicting certain design actions (e.g., Remove Wall, Remove Window), LSTM was not the best model. This work shows that deep learning, particularly LSTM, can be a stepping stone for modeling and predicting sequential decision-making in engineering design and facilitating design automation. By predicting the design process stage at both aggregated level and individual level, the models exhibit designers thinking and strategies. The approach introduced in this chapter is general and can be implemented in many other design areas, especially complex configuration design problems, to extract design decision-making strategies and design heuristics.

However, there are some limitations to our approach. For example, the accuracy we obtained in this study is below the state-of-the-art accuracy of deep learning methods in other fields. This is because design activities are complex and it is challenging to learn prominent patterns due to the heterogeneities within the training dataset. This is different from other types of human behaviors, such as consumer behaviors, where individual shopping mode shows more tractable patterns and would be more easily to be learned by deep neural networks. Additionally, indifferent from other fields where a large amount of human behavioral data can be obtained for model training, such as the customers shopping records and purchase history collected from Amazon, the amount of data collected from human-subject experiments based on students is not ideal. In the next chapter, we develop methods by integrating designers' related attribute with the deep learning model to improve the prediction accuracy.

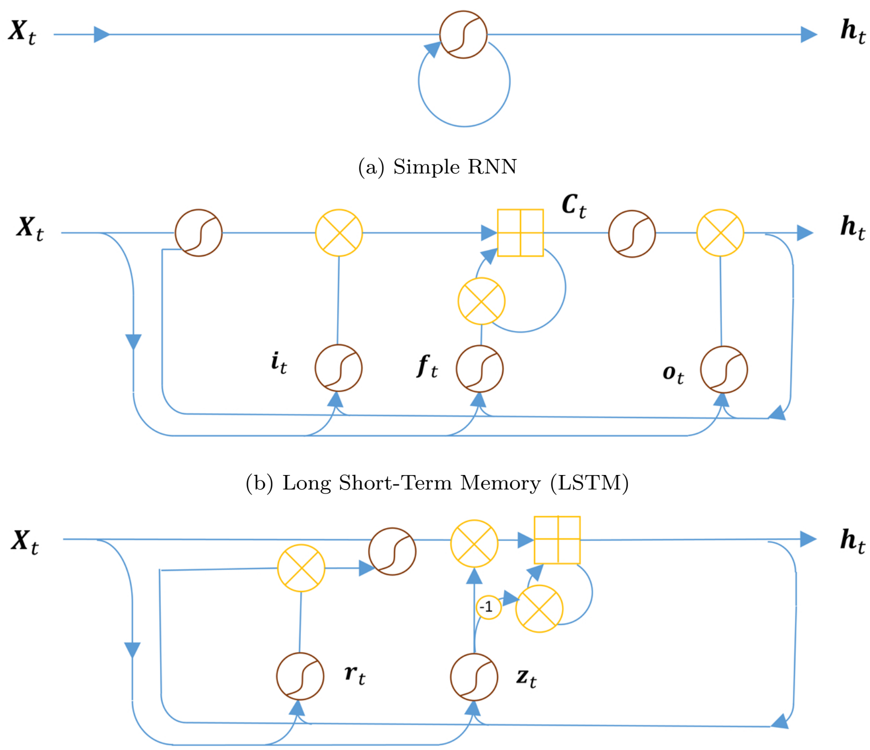
8 Combining Static and Dynamic Data for Predicting Sequential Decision Making

8.1 Overview

Computational modeling of human sequential design process and successful prediction of future design decisions are fundamental to design knowledge extraction, transfer, and the development of artificial design agents. However, it is often difficult to obtain designer-related attributes (static data) in design practices and the research based on combining static and dynamic data (design action sequences) in engineering design is still under-explored. This chapter presents an approach that combines both static and dynamic data for human design decision prediction using two different methods. The first method directly combines the sequential design actions with static data in a recurrent neural network (RNN) model, while the second method integrates a feed-forward neural network that handles static data separately, yet in parallel with RNN. This study contributes to the field from three aspects: a) we developed a method of utilizing designers' cluster information as a surrogate static feature to combine with a design action sequence in order to tackle the challenge of obtaining designer-related attributes; b) we devised a method that integrates the Function-Behavior-Structure design process model with the one-hot vectorization in RNN to transform design action data to design process stages where the insights into design thinking can be drawn; c) to the best of our knowledge, it is the first time that two methods of combining static and dynamic data in RNN are compared, which provides new knowledge about the utility of different combination methods in studying sequential design decisions. The approach is demonstrated in two case studies on solar energy systems design. The results indicate that with appropriate kernel models, RNN with both static and dynamic data outperforms traditional models that only rely on design action sequences, thereby better supporting design research where static features, such as human characteristics, often play an important role.

8.2 Technical Background

In this section, we first provide an overview on the technical background of the RNN. Next, we present the approach that combines both designers' static information and dynamic design process data to model and predicts human sequential decisions in engineering design. To realize such a combination, two methods are developed and these methods are introduced in Section 8.2.2.



(a) Simple RNN

(b) Long Short-Term Memory (LSTM)

(c) Gated Recurrent Unit (GRU)

Figure 8.1: The structures of RNN, LSTM, and GRU

8.2.1 Technical background of Recurrent Neural Network

RNN is a class of deep neural networks, which consists of artificial neurons with one or more feedback loops [153] designed for pattern recognition in sequential data. A typical RNN consists of three layers: an input layer, a recurrent hidden layer, and an output layer. The feedback loop in the recurrent hidden layer enables the RNN to keep the memory of past information. Figure 8.1a shows a standard RNN structure. Given a sequential data with T time steps $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t, \cdots, \boldsymbol{x}_T\}$, at time step t, the hidden state of the RNN is computed by:

$$\boldsymbol{h}_t = f_{RNN}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t), \tag{8.1}$$

where $\boldsymbol{x}_t \in \boldsymbol{R}^N$ is the vector of t^{th} input; $\boldsymbol{h}_{t-1} \in \boldsymbol{R}^M$ is the hidden state of the last time step. Hence, by taking the previous hidden state $\boldsymbol{h}_{t-1}$ and the up-to-date information $\boldsymbol{x}_t$ as input, at each time step, the hidden state $\boldsymbol{h}_t$ encodes the sequence information up to $\boldsymbol{x}_t$. Then, given the hidden state $\mathbf{h}_t$, the output layer is computed as

$$\hat{\boldsymbol{y}}_t = g(\boldsymbol{W}_O \boldsymbol{h}_t + \boldsymbol{b}_O), \tag{8.2}$$

where $\hat{\boldsymbol{y}}_t$ is the predicted output at t^{th} step. $\boldsymbol{W}_O \in \boldsymbol{R}^{K\times M}$ and $\boldsymbol{b}_O \in \boldsymbol{R}^{K}$are the model parameters as well which will be updated during training. $g(\cdot)$ is the output function that adopts the sigmoid function for binary classification and softmax function for multiclass classification. The objective of RNN is to make the predicted sequence close to the observed (true) sequence. By using the observed data as the supervised signal, RNN is trained via the backpropagation through time algorithm [154]. However, the classical RNN model suffers the problem of vanishing gradient [155]. To alleviate this problem, two variants of RNN, i.e., LSTM [156] and GRU [155], are proposed by incorporating the gate mechanism.

LSTM contains special units called memory blocks in the recurrent hidden layer. Each memory block contains an input gate, a forget gate, and an output gate to control the flow of information (see Figure 8.1b). LSTM uses the following equations as the form of f_{RNN} defined in Equation 8.1 to compute the hidden state $\mathbf{h}_t$:

$$\begin{aligned}
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}_f\boldsymbol{x}_t + \boldsymbol{U}_f\boldsymbol{h}_{t-1} + \boldsymbol{b}_f),\\
\check{\boldsymbol{c}}_t &= tanh(\boldsymbol{W}_c\boldsymbol{x}_t + \boldsymbol{U}_{\check{c}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\check{c}}),\\
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}_i\boldsymbol{x}_t + \boldsymbol{U}_i\boldsymbol{h}_{t-1} + \boldsymbol{b}_i),\\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \check{c}_t,\\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}_o\boldsymbol{x}_t + \boldsymbol{U}_o\boldsymbol{h}_{t-1} + \boldsymbol{b}_o),\\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot tanh(\boldsymbol{c}_t),
\end{aligned} \tag{8.3}$$

where $\boldsymbol{x}_t$ is the input at t timestep; $\odot$ indicates the element wise product; $\boldsymbol{i}_t$, $\boldsymbol{f}_t$, $\boldsymbol{o}_t$ are the input gate, forget gate, and the output gate, respectively; $\boldsymbol{c}_t$ is the cell state vector; $\boldsymbol{W}$, $\boldsymbol{U}$ and $\boldsymbol{b}$ are the model parameters.

Another variant of classic RNN is GRU that has fewer parameters compared with LSTM. Figure 8.1c shows a GRU unit. GRU adopts the following equations as the form of f_{RNN} defined in Equation 8.1 to compute the hidden state $\mathbf{h}_t$:

$$\begin{aligned}
\boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r\boldsymbol{x}_t + \boldsymbol{U}_r\boldsymbol{h}_{t-1} + \boldsymbol{b}_r),\\
\boldsymbol{z}_t &= \sigma(\boldsymbol{W}_z\boldsymbol{x}_t + \boldsymbol{U}_z\boldsymbol{h}_{t-1} + \boldsymbol{b}_z),\\
\tilde{\boldsymbol{h}}_t &= tanh(\boldsymbol{W}_h\boldsymbol{x}_t + \boldsymbol{r}_t \odot \boldsymbol{U}_h\boldsymbol{h}_{t-1} + \boldsymbol{b}_h),\\
\boldsymbol{h}_t &= \boldsymbol{z}_t \odot \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t) \odot \tilde{\boldsymbol{h}}_t,
\end{aligned} \tag{8.4}$$

where $\boldsymbol{r}_t$ is the reset gate, while $\boldsymbol{z}_t$ indicates the update gate. It decides how much of the previous information shall be kept.

8.2.2 The research approach

In this sub-section, we present the approach that integrates recurrent neural networks (RNN) as well as clustering techniques to combine both static and dynamic data for predicting human sequential design decisions.

Using RNN to predict sequential design decisions

The proposed approach consists of three parts, data pre-processing, data analysis and preparation, and modeling, machine learning & prediction, as shown in Figure 9.1. In the pre-processing step, we collect raw design behavioral data, which reflect designers' sequential decisions. The raw data can be collected from different sources such as CAD logs [72], design journal of the engineering design process, and documents/sketches generated from thinking aloud protocol studies [157]. The raw data includes a detailed description of design processes generally including both design actions (or design operations) at each time step and associated values of design parameters at that moment. For example, in this study, we focus on the action taken at each step, such as adding a component, deleting a component or changing the parameters of the components, thereby that sequential decision information is extracted from the raw data.

With the collected sequential data of design actions, a design process model is applied to convert design actions into particular design process stages. This proposed step is important and necessary for two reasons: First, the design process model generalizes the design actions to understand designers' behaviors. In the system design task, designers use various design actions which are actually the reflection of design thinking process. Moreover, design actions in different design tasks are different. A design process model at the ontological level (e.g., the FBS model) captures the context-independent essences of design thinking regardless of a particular design task being studied. That helps better probe into designers' thought processes and decision-making in systems design [158]. Second, system design requires a large variety of actions to complete the task. When we want to computationally model these large design actions, it creates a high dimensionality of data. As a consequence, the vectorization of those text data of design actions, e.g., one-hot vectorization, becomes ineffective as the resulting matrix will be very sparse. The design process model at a higher level of description can reduce the dimensionality significantly.

After obtaining the sequential data characterized as design process stages, we adopt RNN to model the decision-making process and predict what the next design process stage that a designer would enter. But before the step of modeling, on one hand, the sequential decision data of design process stages are treated as dynamic data and fed into RNN. On the other hand, the sequential decision data will be used to cluster designers into different groups, where each group contains the

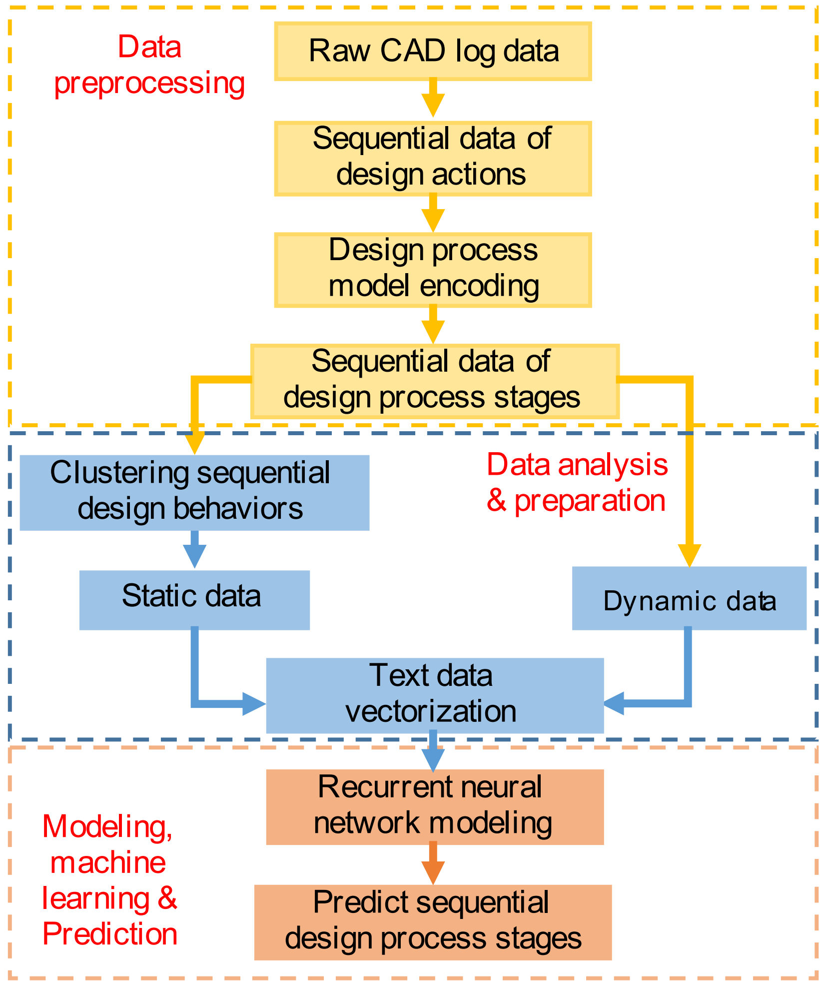


Figure 8.2: The approach of combining static data and dynamic data in RNN to predict sequential design decisions

designers sharing similar decision-making behaviors [159]. The resulting each cluster category is an aggregated reflection of the attributes (e.g., knowledge background, age, etc.) of designers in the same cluster that collectively form their cognitive skills and thinking which, in turn, inform their sequential design decisions. The clustering indexes will be used as the static data input of the RNN. It's worth noting that our approach does not limit researchers to only use clustering information as static data. In this study, we propose to use clustering for generating static data with the motivation of addressing data scarce issues due to limited access to designers' demographics in regular design activities. Researchers can always append additional designers' attributes as the static behavioral feature information to further enhance the model performance. Then, both the clustering information and the sequential design process information are used to train the RNN models.

Finally, the trained RNN models will be able to predict the next design process stage that a designer would enter. We consider the prediction of design process stage as a multi-class classification problem, where each design process stage is a class in our model. We aim to accurately predict the next design process stage given the historical input. The results will be compared and assessed against baseline models through prediction accuracy. In the following subsection, we introduce two methods of combining the static and dynamic data in RNN.

Two methods of integrating static and dynamic data in RNN

First method: Direct input

In the first method, we combine static data with the dynamic data as one single input to the RNN via the concatenation operation. Figure 8.3(a) shows the structure of the first method. Formally, given a designer i, let $\boldsymbol{x}_i$ denote the static data of the designer and $\boldsymbol{x}_t$ indicate the design stage at the t^{th} step of that designer. The concatenation of the static information and dynamic information is represented as:

$$\boldsymbol{x}_{t,i} = [\boldsymbol{x}_t, \boldsymbol{x}_i]. \tag{8.5}$$

To compute the hidden state of the network, we pass $\boldsymbol{x}_{t,i}$ as input to RNN,

$$\boldsymbol{h}_{t,i} = f_{RNN}(\boldsymbol{x}_{t,i}, \boldsymbol{h}_{t-1}), \tag{8.6}$$

where in our experiments, LSTM or GRU is adopted as f_{RNN} to model the user decision making sequences. Finally, we adopt the $softmax$ function as the output layer to predict the next design process stage at time step $t+1$. The $softmax$ function outputs a vector of probabilities for each design process stage. After training, we choose the design process stage with the highest probability as the predicted design process stage. The $softmax$ function is defined as:

$$P(\hat{\boldsymbol{y}}_{t+1,i} = k|\boldsymbol{h}_{t,i}) = softmax(\boldsymbol{h}_{t,i}) = \frac{exp(\boldsymbol{w}_k\boldsymbol{h}_{t,i} + \boldsymbol{b}_k)}{\sum_{k'=1}^{K} exp(\boldsymbol{w}_{k'}\boldsymbol{h}_{t,i} + \boldsymbol{b}_{k'})}, \tag{8.7}$$

where K is the number of design process stages; $\hat{\boldsymbol{y}}_{t+1,i}$ is the predicted design process stage of the i^{th} user at time $t+1$; $\boldsymbol{w}_k$ and $\boldsymbol{b}_k$ are the parameters of $softmax$ function for the k^{th} class.

Second method: Indirect input

In the second method, as shown in Figure 8.3(b), we use the static and dynamic data separately for the model input. The key idea is to first model the static data through an FNN and the dynamic data through an RNN. Since the hidden states of FNN and RNN capture the information of user static information and dynamic design decision data separately, we adopt the concatenation operation to combine the hidden states of both FNN and RNN so that the concatenated hidden

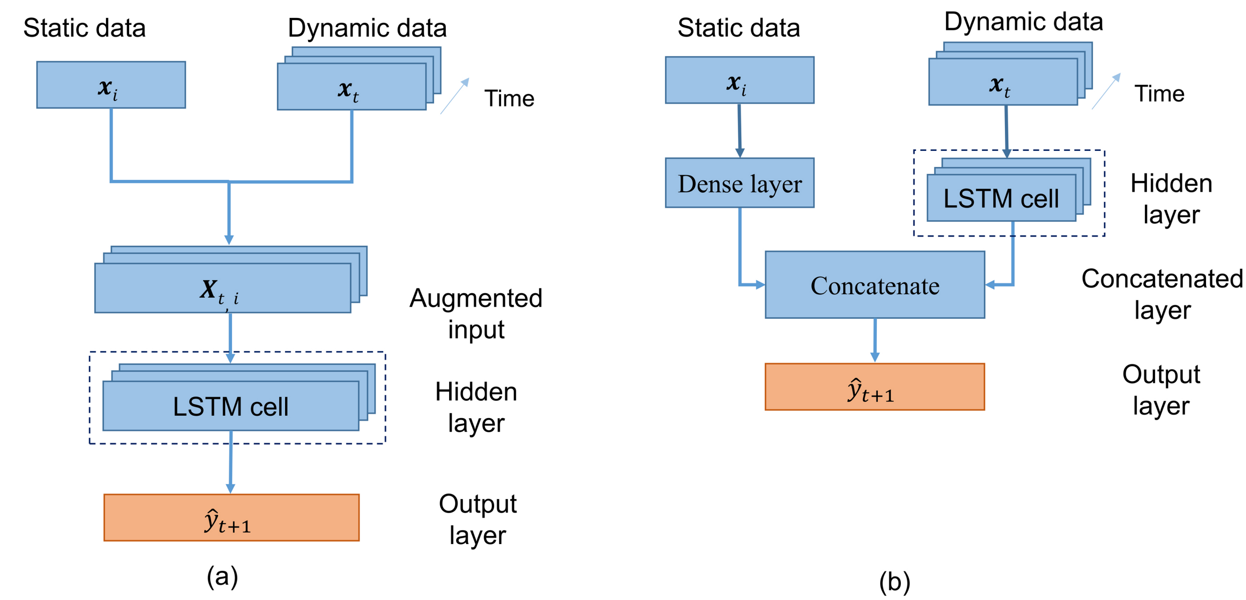


Figure 8.3: a) First method: Direct input b) Second method: Indirect input

state encodes both static and dynamic data information. Finally, we adopt the concatenated hidden state to predict the next design stage. Specifically, the hidden state of the static data can be represented as follows:

$$\boldsymbol{h}_i = f_{FNN}(\boldsymbol{x}_i), \tag{8.8}$$

where f_{FNN} represents a feed-forward neural network. The hidden state for the dynamic input is calculated by the RNN:

$$\boldsymbol{h}_t = f_{RNN}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}). \tag{8.9}$$

In this method, we combine the hidden states of static and dynamic data via concatenate operation:

$$\boldsymbol{h}_{t,i} = [\boldsymbol{h}_t, \boldsymbol{h}_i]. \tag{8.10}$$

Since $\boldsymbol{h}_{t,i}$ captures the hidden information of both static and dynamic data, similar to the first method, we adopt the Equation 8.7 to predict the next design process stage. In this work, we consider the prediction of the next design process stage as a classification task. Hence, we adopt the categorical cross-entropy method [160] as the loss function to train the neural networks:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\boldsymbol{y}_{t,i} log(\hat{\boldsymbol{y}}_{t,i}), \tag{8.11}$$

where N is the number of the users, T is the length of sequence, $\hat{\boldsymbol{y}}$ is the predicted design process stage and $\boldsymbol{y}$ is the actual stage.

8.3 Predicting Sequential Design Decisions in Solar Systems Design – Two Case Studies

In this section, we introduce the design challenges for data collection and data processing procedures for implementing the proposed approach in the context of solar energy system design. The sequential data of design actions are collected from two design challenges. In the first challenge, the task is to design an energy-plus home, while in the second challenge, the task is to design a solarized parking lot at the University of Arkansas. These two design problems exhibit different levels of design complexity [161]. For example, the energy-plus home design is more complex in a sense that it has more design variables and more complex couplings between variables than those of the parking lot design problem. Therefore, they are useful in testing the generality of the proposed approach and methods.

Table 8.1: Design requirements of the two design Problems

Design challenges	Design variables	Design constraints
Energy-plus home design	Story	1
	Number of windows	>4
	Size of windows	>1.44 m^2
	Number of doors	1
	Size of doors ($Width \times Height$)	>$1.2m \times 2m$
	Height of wall	>2.5 m
	Distance between ridge to panel	>0
Solarized parking lot design	Base height	$\geq$ 3.5 m
	Tilt angle	$\leq$ 20
	Solar panel rack	Shall not produce any hindrance to the pedestrian zone and drive ways
	The pole of rack	Shall be placed along with the parking lot line marker

8.3.1 The Design Challenges

The energy-plus home design and the solarized parking lot design

In the energy-plus home design, the objective is to maximize the annual net energy (ANE) of a home with a budget of $200,000. The problem is well-defined meaning that both the design objective and constraints are given. Note that the conceptual part of this design is still an open problem. Therefore, designers need to go through almost the whole design process from conceptual design to embodiment design, and to analysis and evaluation. Figure 8.4(a) shows an example of the energy-plus home design accomplished by one of the participants.

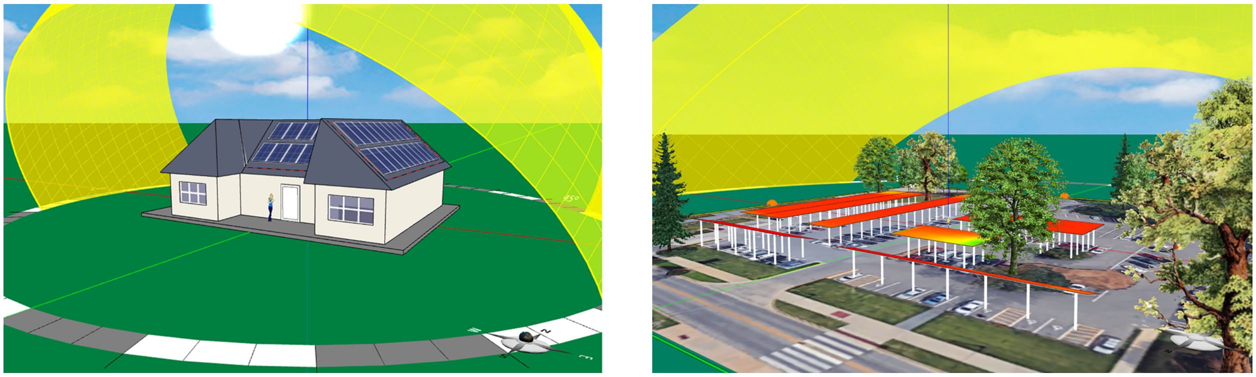
In the solarized parking lot design, the context becomes more authentic. Participants are asked to solarize the Bud Walton Arena parking lot at the University of Arkansas, as shown in Figure 8.4(b). In this design, the objective is still to maximize the ANE, but the budget is $1.5M. Similarly, we provide participants with design requirements. The design requirements of both of the design problems are shown in Table 8.1.

Both design tasks are conducted in Energy3D, a CAD software for solar energy systems, developed by the co-author [111] and is recently extended to a research platform for design thinking studies [2]. Energy3D has several features that are particularly useful to design research. For example, Energy3D logs every design action and design snapshots (the CAD models, not images) in a fine-grained resolution. These data represent the smallest transformation possible on a design artifact. So, the design process can be entirely reconstructed without losing any important details. Energy3D has built-in modules of engineering analysis and financial evaluation that can support the full cycle design of a solar energy system seamlessly. This supports the data collection during both intra- and inter-stage design iterations. Energy3D stores data in a standard data format as JavaScript Object Notation (JSON), which facilitates the post-processing and analysis. The rich data obtained in time scale is essential to the training of the deep-learning model.

In a JSON file, one action is logged in one line. The information contains the timestamp, the design action and its corresponding parameters such as the coordinates of the object, the ANE output, construction cost etc.

The human-subject experiment

The experiment is conducted as a form of design challenge because there is a competition mechanism to incentivize the participants to explore the design space as much as possible. Specifically, the mechanism relates the final reward of a participant to the quality of his/her own design. One design challenge consists of three phases: pre-session, in-session, and after-session. The



(a) (b)

Figure 8.4: Design examples from one participant: (a) Energy-plus home design (b) Solarized parking lot design

pre-session lasts about 30 minutes and is designed for familiarizing students with the Energy3D operating environment, the design problem, and basic solar science concepts. The purpose is to mitigate the effect of the learning curve and minimize the potential bias caused by different levels of pre-knowledge of participants. The data generated in the pre-session are abandoned and not used in the case study. During the in-session, participants work on the design problem based on the instruction, including the design statement and the requirements aforementioned. A record sheet is also provided for participants to record the design objective values for feedback and reinforcement of decision-making. The in-session lasts about 90 minutes. The after-session, which lasts about 10 minutes, is for participants to claim rewards and sign out the challenge.

In the energy-plus home design, 52 students from the University of Arkansas participated. The participants are indexed based on their registered sessions and laptop numbers. For example, A02 indicates a participant who was in session A and used laptop #2. On average, each participant spends 1500 actions (CAD operations) to accomplish the task. There are some actions, such as adjusting camera view, adding human, that do not essentially affect the design artifact, are removed. With this treatment, each participant uses 335 valid design actions on average among which 115 actions are unique. In the solarized parking lot design, there were 41 participants. After removing trivial actions, each participant uses an average of 350 design actions, among which 72 actions are unique.

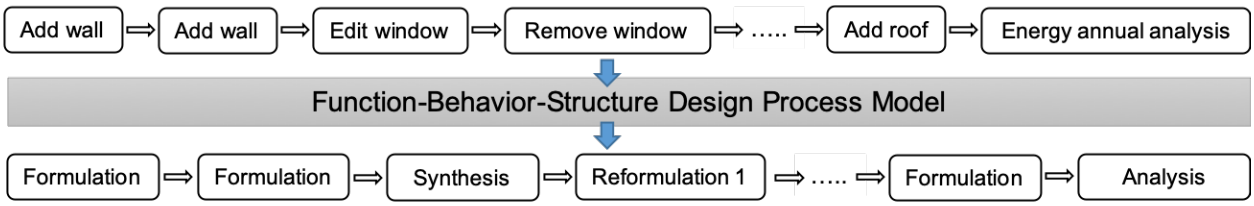


Figure 8.5: Transformation of the sequential data of design actions to the sequential data of design process stages based on FBS design process model

8.3.2 Data preparation and combination of static and dynamic data

Transformation of design action sequence to design process stages

After obtaining the raw sequential CAD log data, the design actions, such as “Add rack" and “Add solar panel", are extracted as the sequential data for training the RNN models. As introduced in Section 8.2.2, a design process model is needed to transform the design action sequence to the sequence of design process stages for better understanding designers’ thinking and decision-making at a higher level of cognition. In this study, the FBS-based design process model [162] is adopted and the coding scheme in Table 8.2 is used to categorize each design action into one of the seven design process stages, including Formulation (F), Synthesis (S), Analysis (A), Evaluation (E), Reformulation 1 (R1), Reformulation 2 (R2), and Reformulation 3 (R3). The FBS-based process model is adopted as it is a well-accepted design ontology that can be used universally to represent a systems design process regardless of the application context [33]. Also, it is evident that the FBS ontology well represents design thinking strategies with its constructs that capture the actions to be taken for achieving design objective and evaluating design performance [163]. Figure 8.5 shows an example of one segment of a design sequence from a participant and its transformation to the design process stages after encoding with the FBS model.

Use clustering analysis to obtain static behavioral data

With the sequential data of the design process stages, we perform the clustering analysis to categorize participants into different classes representing different sequential decision-making behavioral patterns. This analysis helps to obtain the static data pertaining to each designer.

In this study, the Markov chain-based approach proposed in our previous study [159] is adopted for the clustering analysis. As a quick summary, the first-order MC model is applied on each designers’ sequential data to obtain the 7×7 transition matrix, which is then converted to a 49×1 vector. By combining the transition vectors of all the N participants’, we obtain a $49 \times N$ matrix that will be used for clustering analysis. In total, we test three clustering methods including K-means clustering, hierarchical agglomerative clustering and network-based clustering, and each method is tested with three options of the number of clusters, i.e., 4, 5 and 6, which are suggested by the Elbow plot [164]. Based on the metric of effectiveness defined in [159], it is found that for the energy plus home design dataset, hierarchical agglomerative clustering with 4 clusters performs best while for the solarized parking lot design dataset, network-based clustering with 5 clusters measured performs best. For the details of clustering analysis approaches and their implementation, please refer to [159]. Figure 8.6 shows an example of the results of hierarchical agglomerative clustering with 4 clusters for the energy-plus home design dataset.

The clustering analysis gives every designer an index that is used as the static information to be combined with the dynamic data (i.e., the sequential data of design process stages) for training the RNN models. In the following subsection, we introduce two different methods of realizing such a combination.

Combining static and dynamic data

Before combining the static and dynamic data for RNN modeling, we transform the static and dynamic text data to one-hot vectors because neural network models cannot work with text data directly. Since the FBS model has already helped reduce the dimensionality of sequential

Table 8.2: The FBS model and the proposed coding scheme for design actions [2]

Design process	Definition and interpretation	Types of design action
Formulation	Generate Function from Requirement and from Function to Expected Behavior.	Add any components
Analysis	The process generated from Structure.	Analysis of ANE
Synthesis	Generate and tune Structure based on the Expected Behavior.	Edit any components
Evaluation	The comparison between the Expected Behavior and the behavior enabled by the actual structure.	Cost analysis
Reformulation 1	The transition from one structure to a different structure.	Remove structure
Reformulation 2	The transitions from Structure to Expected Behavior.	Remove solar device
Reformulation 3	The transition from Structure to Function.	Remove other components

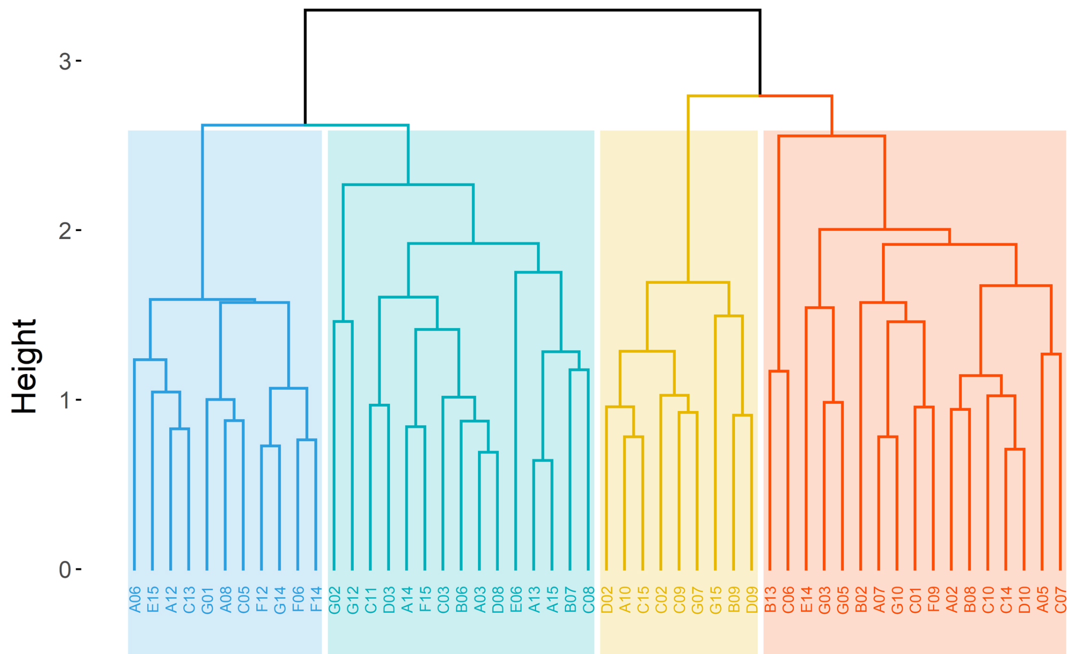


Figure 8.6: Hierarchical clustering of four groups for the energy-plus home design dataset. X-axis label indicates the participants who are clustered together in different colored boxes.

data from the action level to the process stage level, one-hot vector under this circumstance is an appropriate and efficient method to vectorize the data.

In the first method, we combine the static data directly with the dynamic data, as shown in the schematic diagram in Figure 8.7. In this example, the designer is from cluster 2 (identified by the hierarchical agglomerative algorithm) and its corresponding one-hot vector is appended behind each one-hot coded sequential design process stages. Since the cluster index does not change over time, the same one-hot vectors are appended as long as the sequential data are for the same person. Then, the combined vectors are fed into the RNN layers as same as what a normal RNN training process does.

Different from the first method, the second method allows static data and dynamic data to be processed in separate layers. This means the cluster data and the sequential data are handled separately during the input and are trained by different models, and the resulting outputs from the hidden layer are then combined before sending to the output layer for backpropagation. For example, as shown in Figure 8.8, the one-hot vector of the cluster index of number 2 is passed as the input of the hidden layer, i.e., an FNN model. At the same time, the one-hot encoded sequential data are passed as the input of the hidden layer, i.e., an RNN with LSTM layers. Then, the results from both the FNN layer and the LSTM layer are combined and then passed to the output layer

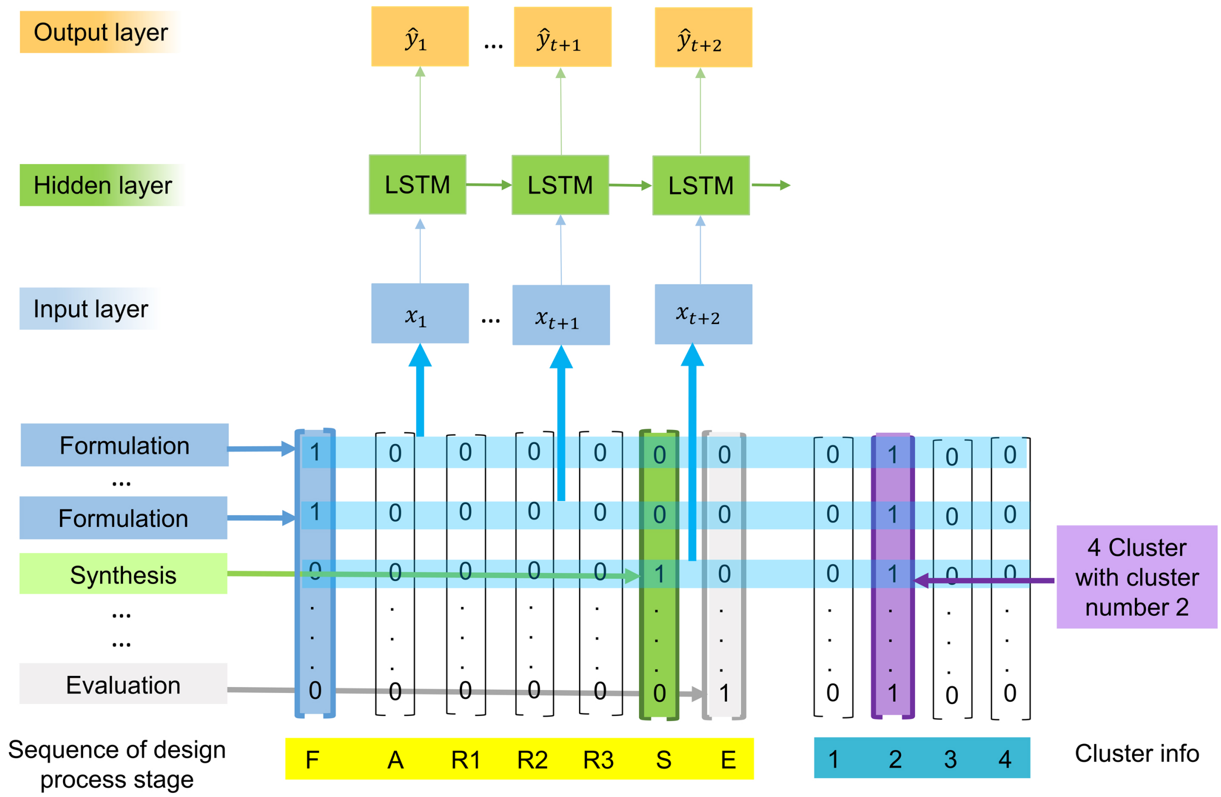


Figure 8.7: The cluster information is added directly to the sequential data of a designer who is in Cluster #2

for training.

With both methods, we train the models and then use the trained model to predict the next design process stage. In the following section, we compare the results of the prediction accuracy from both methods. We also perform the sensitivity analysis to investigate how the prediction accuracy would change with different model configurations and experimental settings.

8.4 Model Implementation and Evaluation

In this section, we first present the model setup and the method of evaluating the models' predictive performance. Then, the results are of the prediction accuracy for both methods in two case studies are presented and discussed.

8.4.1 Mode setup and evaluation method

Baseline: The RNN models that use the sequential data of design process stages only without using static data are chosen as the baseline model for comparison and evaluation.

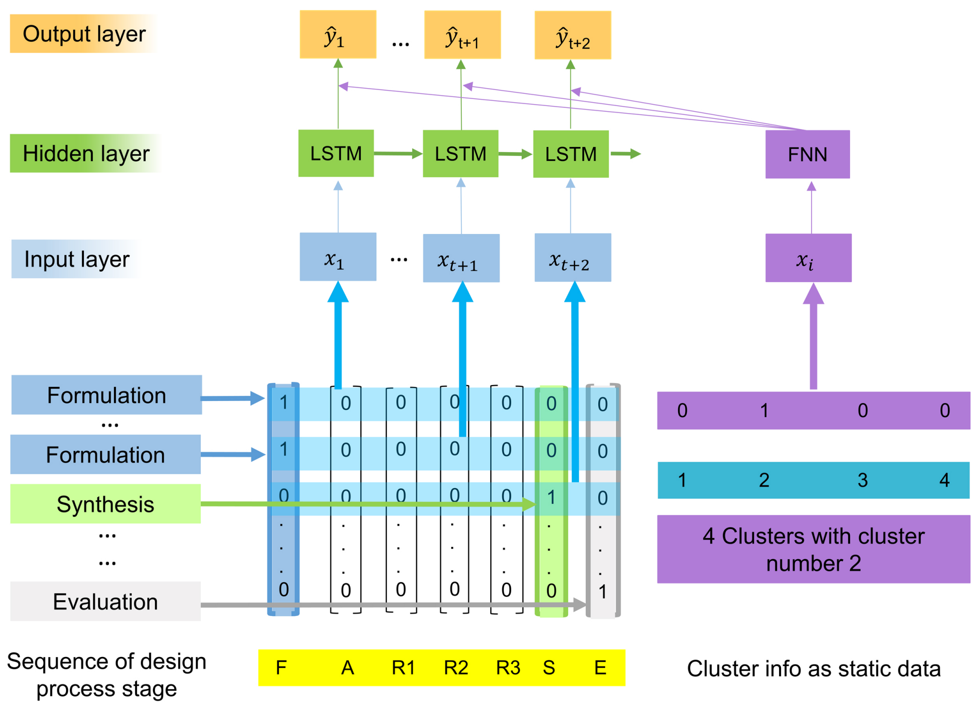


Figure 8.8: Combining the cluster data into the sequential data in separate layers. Cluster data are the input of an FNN layer and sequential data are the input of an LSTM layer.

Cross-validation: We conduct k-fold cross-validation [165] to evaluate the performance of the models. In the k-fold cross-validation, the data set is split into k partitions, $\{C_1, C_2, \ldots, C_k\}$, where k is the total number of validation folds. Then, k rounds of training and testing are performed in a way that in each iteration the model is trained on partitions $\{C_1, \ldots, C_{i-1}, C_{i+1}, \ldots, C_k\}$ and evaluated on C_i partition. In this study, we adopt the 4-fold cross-validation technique to evaluate the models. In order to split the dataset into 4 folds, two different split procedures are used as there are two different data combination methods. In the first method, we directly add cluster information to the corresponding designers' sequential data. Then we split each dataset into 4 folds. In the second method, we split both the sequential data and cluster data into 4 folds, separately.

Hyperparameter: In order to find the best hyperparameter settings of the RNN models for training, we trial and error different settings and choose the best one to measure the accuracy. As we compare those models with the baseline models, we run different settings on the sequential data without combining the cluster information first in order to obtain the best configuration. Then, these settings are adopted in the models combining both static and dynamic data for a fair comparison.

We implemented the models using LSTM and GRU, respectively. The dimension of the hidden layer is 256. To prevent over-fitting, we use 20% dropout regularization [166]. We trained each of the settings for 50 epochs although after 30 epochs the accuracies are not significantly improved. Adaptive moment estimation (Adam) is adopted as the stochastic optimization algorithm for the estimation of parameters in backpropagation [167]. The computation is performed with Keras deep-learning library [147] while TensorFlow [168] is running as the back-end.

The metrics for predictive performance: As previously mentioned, we get the prediction of t^{th} design process stage using the previous $t-1$ design process stages. Therefore, for a design sequence with n actions, the model will produce $n-1$ predictions. These predictions are compared with the observed data and the total number of correct predictions (n_i^{cp}) can therefore be obtained. The prediction accuracy can be therefore defined in Equation (8.12) by average the scores from each round of the cross-validation.

$$Prediction\ Accuracy = \frac{1}{R}\sum_{i=1}^{R}\left(\frac{n_i^{cp}}{n_i^{max}-1}\right), \tag{8.12}$$

where R is the number of rounds (i.e., iterations) in the cross-validation and n_i^{max} is the length of the longest design sequence in the round i. In this paper, only the prediction accuracy of the testing data (i.e., the testing accuracy) is reported. To account for the uncertainties, we conduct 4-fold cross-validation twice for each of the four models. As a result, in total, we obtain eight results of the prediction accuracy for each model. Besides the prediction accuracy, we also adopt the area under the receiver operating characteristics curve (AUROC) [169]. AUROC is measured by two characteristics, true positive rate (TPR) and false positive rate (FPR). TPR indicates what proportion of a particular class (e.g., Formulation in our case) correctly predicted. FPR indicates the proportion of the other classes (e.g., other design process stages than Formulation) incorrectly predicted. With these two characteristics, the AUROC is computed at different probability thresholds from 0 to 1. However, when computing the accuracy, it is the design stage with the highest probability to chosen as the prediction. so, the prediction accuracy is essentially measured by the ratio between the number of true positive and the total number of prediction at a specific probability threshold. Since it doesn't take the true negative into account, the value should be smaller than either TPR or FPR. Therefore, the defined prediction accuracy is a more strict measurement for the performance evaluation, while AUROC is more comprehensive. Using both metrics together can well reveal the overall predictive performance of the models.

8.4.2 Results and Discussion

Tables 9.2 and 8.4 show the results of the accuracy and AUROC score on the energy-plus home design task and solarized parking lot design task, separately. As we can observe from the tables, all models achieve decent performance considering its a seven-class classification task, which

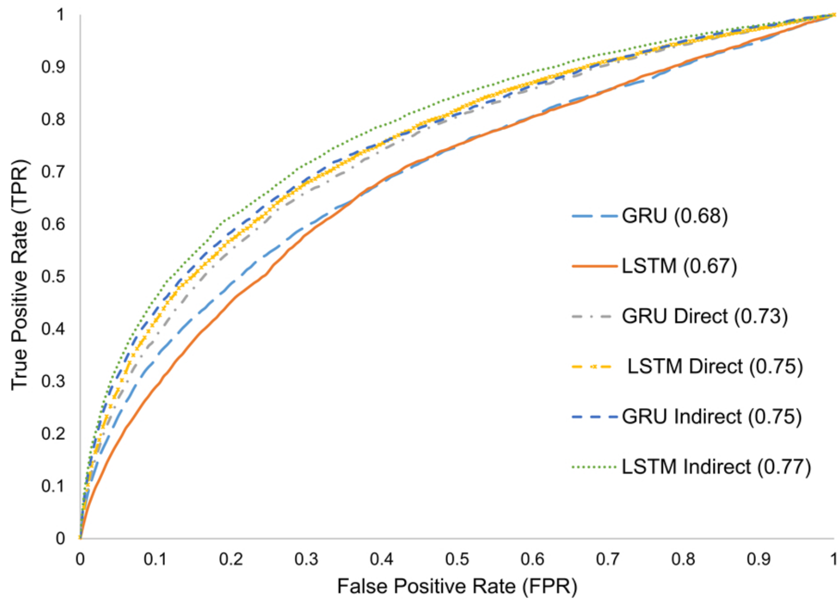


Figure 8.9: The ROC curves of baseline models and the models with static data for energy-plus home design dataset

shows the advantage of deep-learning models for predicting sequential data of design decisions. The results indicate that our proposed two methods outperform the baseline models in both cases, which suggests that incorporating the static data into the LSTM/GRU model can improve the performance of design stage prediction.

Meanwhile, the results of the solarized parking lot design are better than those of the energy-plus home design on average, especially in terms of the AUROC score. As mentioned in

Table 8.3: The testing accuracy and the AUROC scores for the energy-plus home design dataset

Dataset	**Combination method**	**RNN variants**	**Cluster method**	**Testing accuracy**	**AUROC score**
Energy-plus home design	Baseline	LSTM	N/A	58.06 ± 1.69	0.67
		GRU	N/A	58.26 ± 3.13	0.68
	Direct	LSTM	4 clusters using hierarchical clustering	60.51 ± 1.63	0.75
		GRU		58.31 ± 3.49	0.73
	Indirect	LSTM	4 clusters using hierarchical clustering	60.60 ± 1.83	0.77
		GRU		59.57 ± 1.72	0.75

Section 8.3.1, the design complexity of the solarized parking lot design is lower than that of the energy-plus home design. As a result, there is fewer number of unique design actions taken in the former design challenge, thus the occurrence of the same design process stages are more than the number in the energy-plus home design. Therefore, the patterns of design sequences from the solarized parking lot design task could be more easily captured by the neural networks.

In particular, for the energy-plus home design task, it is observed that the large performance gains in terms of AUROC, e.g., an increase of 8% and 10% in LSTM models, for both direct and indirect methods (also see Figure 8.9). Meanwhile, it is noticed that the indirect combination method achieves slightly better performance than the direct method. It indicates that for the energy-plus home design task, combining static data is useful for the design stage prediction, while combining the hidden representations of static and dynamic data as inputs to the classifier is a better option.

For the solarized parking lot design task, the proposed indirect and direct methods still perform slightly better than the baseline models. These result again shows the effectiveness of our proposed deep-learning approach. Especially, as shown in Figure 8.10, the indirect combination method achieves higher AUROC values than the baselines with an increase of 3%. However, compared with baselines, our proposed methods in the solarized parking lot design task do not achieve similar gains as in the energy-plus home design task. The potential reason is that for the solarized parking lot design task, the participants are clustered into 5 groups instead of 4 groups. Then, the number of designers in each cluster for the solarized parking lot design is smaller than that for Energy-plus home design. For example, the smallest cluster in the Energy-plus home design case study has 9 members, while the smallest cluster in the solarized parking lot design only has 4

Table 8.4: The testing accuracy and the AUROC scores for the solarized parking lot design dataset

Dataset	**Combination method**	**RNN variants**	**Cluster method**	**Testing accuracy**	**AUROC score**
Solarized Parking lot design	Baseline	LSTM	N/A	60.91 ± 1.63	0.78
		GRU	N/A	59.59 ± 4.13	0.79
	Direct	LSTM	5 clusters using network-based clustering	61.93 ± 4.08	0.79
		GRU		63.38 ± 5.71	0.79
	Indirect	LSTM	5 clusters using network-based clustering	61.98 ± 1.83	0.82
		GRU		61.12 ± 1.72	0.81

Table 8.5: Statistical t-test on the difference between the prediction accuracy of the baseline models and the models developed in the two case studies

Energy-plus home design			
RNN variants	Hypothesis testing	t score	p-value
LSTM	Baseline vs. Direct method	-4.04	0.0024
	Baseline vs. Indirect method	-4.34	0.0017
GRU	Baseline vs. Direct method	0.26	0.40
	Baseline vs Indirect method	-0.53	0.30
Solarized parking lot design			
LSTM	Baseline vs. Direct method	-5.51	0.00044
	Baseline vs. Indirect method	-3.12	0.008
GRU	Baseline vs. Direct method	-4.27	0.0018
	Baseline vs. Indirect method	-1.26	0.12

members. Hence, it would be difficult for the LSTM/GRU model to identify useful hidden patterns from a smaller number of data points for prediction. Overall, the experimental results indicate that the use of static data can improve the performance of the LSTM/GRU models for design stage prediction, especially when designers have similar design thinking (i.e., same static data), could further generate similar design sequences.

To evaluate the statistical significance of the difference between the developed models and the baseline models, the paired t-test is conducted. The null hypothesis (H_0) is that the mean of the prediction accuracy of the models combining static information is equal to that of baseline models; and the alternative hypothesis (H_a) is that the former is significantly less than the later. Table 8.5 shows the results of the t-test for both case studies. With the level of significance of 0.05, the p-values in the table indicate that in both studies regardless of the combination methods employed, the performance of LSTM model is always significantly better than that of the baseline models; while the performance of the GRU model is only supported in the second case study using the direct combination method.

8.5 Conclusion

In this chapter, we established a research approach based on deep recurrent neural network (RNN) to predict human sequential design decisions. The contributions of this study can be summarized in the following aspects. First, we introduced two methods of combing static and

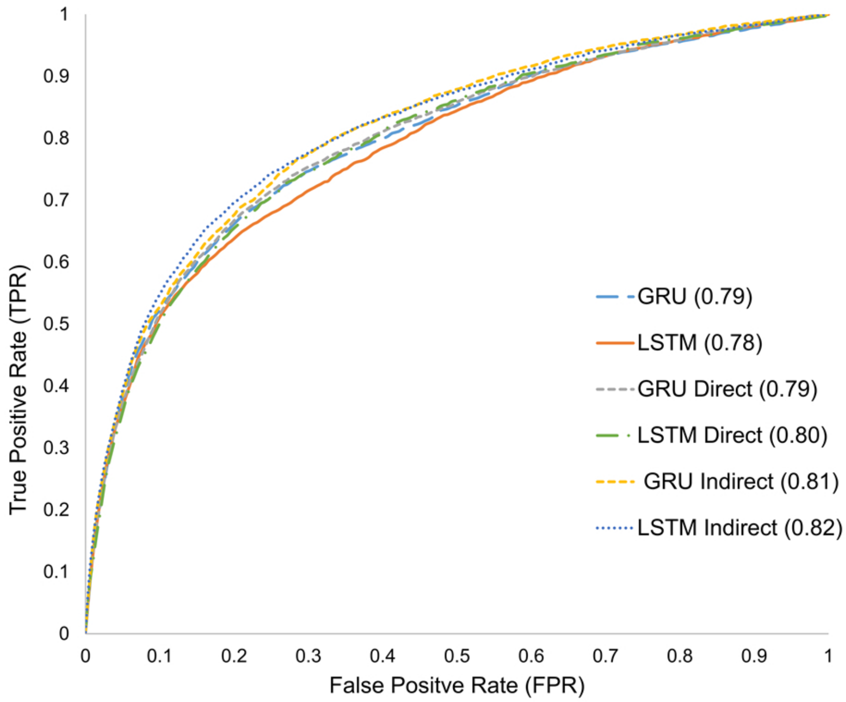


Figure 8.10: The ROC curves of baseline models and the models with static data for solarized parking lot design

dynamic data for sequential design decision prediction. The first one is a new method which directly combines the static and dynamic data as inputs to the RNN, while the second method combines the hidden representations of static and dynamic data, which are derived from the FNN and RNN, respectively, into the classifier. Second, we have developed a novel clustering-based method to derive a surrogate static feature that can eventually enhance the prediction through the integration of unsupervised learning (i.e., the clustering algorithm) and supervised learning (i.e., the RNN models). Third, we developed an approach that integrates the FBS design process model and the one-hot vectorization to transform design actions to design process stages in order to tackle the high dimensionality associated with the design sequence data and draw insights into design thinking. Lastly, to the best of our knowledge, this is the first work of comparing two different methods of combining static data and dynamic data in an RNN-based framework. Therefore, our study does not only provide new knowledge on how well the deep RNN would perform by combining static and dynamic data in an engineering design application, but also provide new knowledge on how well each combination method (i.e., direct input vs. indirect input) would performance with different kernel settings (e.g., LSTM vs. GRU). The performance of our proposed approach and methods are evaluated in two design case studies. The experimental results indicate that with appropriate models, RNN with both static and dynamic data outperforms traditional models that only rely on design action sequences, thereby better supporting design research where static features, such as human characteristics, often play an important role. So far, we have developed several deep learning methods for predicting design actions based on the historical data. However, these methods dose not provide importance of the mostly used or prominent design actions. Therefore, in the next chapter, we develop a design agent based on the reinforcement learning algorithm for predicting design actions.

9 Developing Reinforcement Learning-Based Framework for Understanding the Transferrability of Design Knowledge

9.1 Overview

Although artificial intelligence (AI) exerts its influence on design automation, humans have shown surprising capability in many design problems, for example, transforming ill-defined problems into well-defined problems via requirements analysis and function modeling. Therefore, developing an artificial design agent with the integration of human heuristics that mimic human design behaviors is crucial to advance design automation, human-AI collaboration, and training novice designers in design education. However, developing a design agent requires a large amount of design behavioral data. For many design problems, scarcity of human data is also a big issue. Transferring learned design knowledge from one problem to another can help solve the data scarcity problem. The objective of this chapter is to collect empirical evidence and perform computational assessments to test the transferability of design knowledge between totally different design contexts. To achieve this objective, we first develop a design agent based on reinforcement learning (RL) to mimic human design behaviors. A data-driven reward mechanism based on the Markov chain model is introduced so that it can reinforce prominent and beneficial design patterns. The design knowledge learned from one design problem (source task) is then transferred to another design problem (target task). Using two solar system designs as a case study, one set of data is used to train the design agent, and the other set of data is used to test the transferability of the design agent. The results first show that the RL-based agent provides a prediction accuracy higher than that of a baseline model using the Markov chain in the source task as well as in the target task based on the knowledge learned from the source task. This indicates that the RL agent can successfully capture the design knowledge and also transfer the knowledge. This study corroborates that the transfer of design knowledge is possible not only between different design problems in the same context, but also between problems from two totally different design contexts.

9.2 Technical Background and Research Approach

9.2.1 Preliminaries

Typical RL approaches rely on the formalism of a Markov Decision Process (MDP) to learn optimal behaviors in sequential decision-making problems. The goal of an MDP is to find the optimal policy for decision making based on rewards [170]. Q-learning helps to find such a policy by generating a Q-value for each state-action pair that is used to determine the best decision for a given state of a problem environment. The Q-values of all state-action pairs are typically stored in a Q-table that is learned through multiple iterations in an RL process. At the beginning, we initialize the Q-table with zeros as the state-action dynamics is not known at this phase. In the following steps, the agent selects an action by exploitation or exploration. In the initial phase, the agent does not have much information from the Q-table and mostly explores the action space by taking random actions. The agent updates the Q-table on the basis of the reward it receives from these random actions. Once the agent performs an adequate number of iterations and collects information about the environment, it begins to exploit the action space. This trade-off between exploitation and exploration can be handled by the epsilon-greedy algorithm. Once the agent selects an action, it reaches a new state $S^{'}$. In the new state, the agent selects the best possible action that yields the maximum Q-value and finds the corresponding rewards. Based on the rewards, the Q-value is updated according to the following equation:

$$Q_t(S,A) = Q_{t-1}(S,A) + \alpha(R(S,A) + \gamma \max\{Q(S^{'},A^{'})\} - Q_{t-1}(S,A)), \tag{9.1}$$

where, $Q_t(S,A)$ is the new Q-value for state S and action A in the next iteration t. $Q_{t-1}(S,A)$ is the current Q-values. α is the learning rate – a hyperparameter that defines how much new information can be accepted in the current iteration vs. the old information from previous iterations. When α is close to zero, Q-values are never updated; whereas, an α value close to 1 means that the training process occurs quickly. R(S,A) is the value of the reward for taking action A in the state S. γ is the discount factor that controls how much future rewards will be taken into account when updating the Q-value. $maxQ(S^{'},A^{'})$ is the maximum expected future value. The agent will iterate over multiple steps to update Q-table values till convergence. In this paper, we adopt a probabilistic model for action transition. The agent chooses one of the actions with the following probability function based on the Q-values [171],

$$Pr(a|s) = \frac{\exp(\theta \cdot Q(s,a))}{\sum_{a_i \in A_i} \exp(\theta \cdot Q(s,a_j))} \tag{9.2}$$

where A_i denotes the action space of an agent. The equation takes values from the Q-table and provides a probability of taking each possible action (a) at a given state (s). The hyperparameter $\theta \in [0, \inf)$ determines the decision-making strategy of an agent. When θ is zero, the equation provides a uniform distribution (i.e., all design actions are equally likely to be selected. When θ goes to infinity, the probability of the action with the highest Q-value (e.g., the most frequently occurring design action at a given state) approaches 1. Note that this model is similar to the logit choice model commonly used in the design and marketing literature [172] where Q-values correspond to the utility of discrete choices.

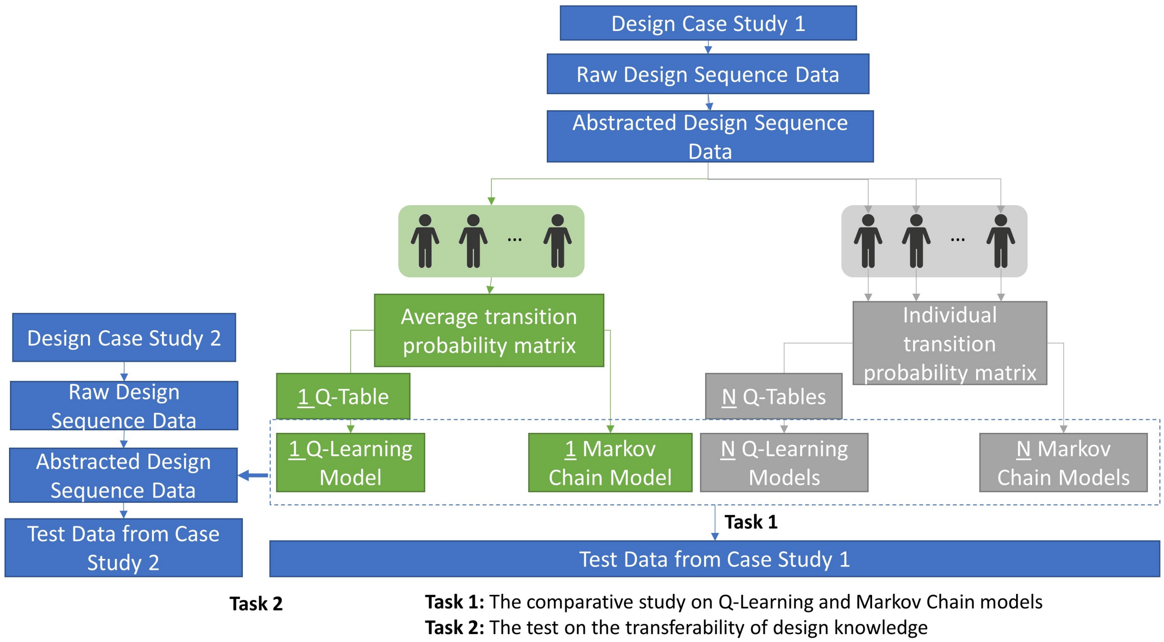


Figure 9.1: The overview of the research tasks

9.2.2 Research Approach

This study consists of two tasks, as shown in Figure 9.1. The first task is develop a RL-based design agent for mimicking human design behavior. The data used to train the agent are obtained from sequential design actions performed by designers. The actions could be adding a component, deleting a component, or changing the parameters of a component. To evaluate the performance of the agent, we conduct a comparative study using Markov chain model as the baseline.

Once sequential design data are collected, a design process model is applied to convert each design action to its corresponding design process stage. A design process model at the ontological level captures the context-independent essence of design thinking regardless of a particular design action involved. Therefore, such an action-to-stage conversion helps generalize design knowledge and facilitate the transfer of design knowledge from one problem to another. Moreover, by applying the design process model to group actions in the stage, the procedure turns out to be a dimension reduction that helps improve the computational efficiency. This is particularly useful in system design, where there could be a large number and a variety of actions involved.

After obtaining the sequential data of the design process stages, our first task is to train Q-learning models to develop the RL-based design agents that mimic human design behaviors. In the Q-learning model, we utilize the transition probability matrix identified from the first-order Markov chain model as the reward table. There are two different ways to obtain the transition probability matrices when creating the reward table. One way is to use the average transition probability matrix that aggregates the sequential design data of N subjects (designers). In this situation, one Q-learning model will be developed to predict the behaviors of N designers. The other way is to use each individual's transition probability matrix to construct N Q-learning models that can be used to predict the design actions of N designers separately. In both ways, we tune the hyperparameter θ and investigate how it influences the accuracy of the Q-learning model in predicting the sequential actions of each individual designer. Based on these configurations, we are interested in knowing which way is the better way to construct the reward table for the RL-based agent. To evaluate the performance of the RL-based agents in prediction, we compare them with those without a reinforcement mechanism, i.e., the models purely based on MCs. The results of these comparative studies are presented in Section 5.

Our second task is to test the transferability of learned design knowledge between design problems. In particular, we apply the Q-tables learned from the source design problem (design problem 1) to predict the designers' behaviors in the target design problem (design problem 2).

9.3 Case study

9.3.1 The design challenges

In this study, the designers' behavioral data are collected from two design challenges: the energy-plus home design and the solarize UARK campus design. We train our design agents using the energy-plus home design dataset. Therefore, it is treated as the source task, and the solarize UARK campus design is used as the target task. In the following, each of the tasks is described in detail.

The energy-plus home design and the solarize UARK campus design

In the energy-plus home design problem, participants were asked to build a solar-powered home in Dallas, Texas. The objective is to maximize annual net energy (ANE) while minimizing construction cost. The overall budget for this design problem is $200,000. In addition, we set specific design constraints to confine the design space, as summarized in Table 9.1. This system design problem involves many design variables with complex coupling relationships among these variables (e.g., designers may want to add many solar panels for higher ANE, however, the distance between solar panels could not be too small, so there is a limit for the number of solar panels to be placed). For this reason, the design space is large, and different designers may take different strategies to explore and exploit the design space.

In contrast to the energy-plus home design problem, the solarize UARK campus design is more open-ended. In this problem, participants were provided with a computer-aided design

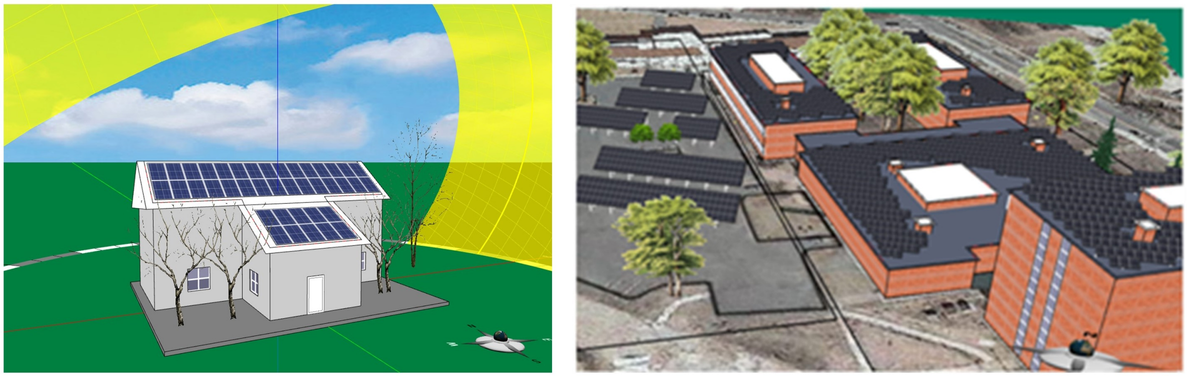


Figure 9.2: An example of the energy-plus home design problem (left) and an example of the solarize UARK campus design problem (right)

(CAD) model of a student housing complex and its adjacent parking lot on a university campus and asked to use the open space on the roof of the buildings and the paved area on the parking lot to design the solar system. The goals of the design challenge are three-fold. First, the annual energy output should be greater than 1,000,000 KWh. Second, the overall budget should not exceed $1,900,000. Finally, the payback period should be less than 10 years. Participants are encouraged to work iteratively and record the performance that different solutions offered, so that they can compare their own design iterations to continuously improve the performance of their designs. To achieve the goals successfully, designers need to smartly control the design variables, such as the location, length, tilt angle, and model of each solar panel. There are dependencies among these variables; therefore, participants will be benefited from systems thinking. For example, the optimal tilt angle of a solar panel depends on the height and where it will be placed. The degree to which designers could manipulate each variable is limited by a set of constraints, as shown in Table 9.1. For example, participants have to arrange the solar panels while maintaining a minimum distance between every two panels, and can only choose one model from three options.

Both design problems are carried out using Energy3D, a computer-aided design (CAD) software for renewable energy systems ([35]; [173]). Energy3D collects design data in non-intrusive way. The non-intrusive data collection process can reduce the cognitive bias during an experiment. Energy3D logs design data at a fine-grained level. In particular, it logs every design action performed and collects design artifacts every 20 seconds. Therefore, the data collected from Energy3D fully capture what designers do (i.e., design actions) throughout the design process. Energy3D collects the design process data in JSON format, which records time stamps, design actions, design artifacts, and simulation results. On average, a participant has about 1500 lines of design process data. An example of two lines of the design actions log is presented in the text box below.

Table 9.1: Design requirements of the design challenges

Design challenges	Design variables	Design constraints
Energy-plus home design	Story	1
	Number of windows	> 4
	Size of windows	$>1.44\ m^2$
	Number of doors	≥ 1
	Size of doors (Width $\times$ Height)	> 1.2 m $\times$ 2m
	Height of wall	>2.5 m
	Distance between ridge to panel	>0
Solarize UARK campus design	Solar panel model	Choose 1 of 3 options
	Solar panel height	$\geq$ 3.5m
	Solar panel width	5.25m - 6m
	Panel Placement (Overall)	Panel edges must not overlap
	Panel Placement (parking lot)	$\geq$ 7m from the nearest panel

{"Timestamp": "2020-05-23 08:17:38", "File": "Design-Contest.ng3", "Edit Rack": {"Type": "Rack", "Building": 2, "ID": 485, "Coordinates": [{"x": 1.496, "y": 47.053, "z": 54.7}]}}
{"Timestamp": "2020-05-23 08:19:49", "File": "Design-Contest.ng3", "PvAnnualAnalysis": {"Months": 12, "Panel": "All", "Solar": {"Monthly": [892.25,1060.33,1478.38,1544.75,1819.32,1950.18,2048.8, 1876.89,1423.77,1241.81,794.41,697.7], "Total": 511869.84}}}

In this study, we extract only design actions related to design objectives, such as 'Add wall', 'Edit wall', 'Edit roof', 'Show sun path', etc. We ignore design actions that have no effect on design outcomes, such as 'Camera' and 'Add tree.' This post-processing leads to 115 unique design actions in the energy-plus home design problem and 106 unique design actions in the solarize Uark campus design problem.

The human-subject experiment

Both of the design experiments are conducted as a form of design challenge as it motivates the participants to explore the design space as much as possible and find optimal solution. Additionally, the designers are incentivized by monetary reward which relates to the quality of his/her own design. The Energy-plus home design problem is conducted in class setting and consisted in three phases: pre-session, in-session and after-session. In the pre-session, the designers get familiar with the Energy3D environment, the design problem and basic solar science concepts. The guide in the pre-session mitigates the effect of the learning curve and minimize the potential bias caused by different levels of pre-knowledge of participants. The pre-session lasts about 30 minutes while the in-session lasts about 90 minutes. During the in-session, participants perform the design task according to the design requirements. The after-session, which lasts about 10 minutes, is for participants to claim rewards and sign out the challenge. The solarize UARK design problem is conducted on the virtual settings. At the begining of the design challenge, participants are given all the necessary information and short-tutorial session through a introductory presentation. The participants are given seven full days to complete the design task on their own time.

In the Energy-plus design problem, A total of 52 designers from the University of Arkansas participated in the design challenge. The participants are indexed according to their registered sessions and the laptop numbers. Sessions are indexed by the letters from A to G and laptops are indexed with numbers. For example, A02 indicates that the participant joined at session A and worked on the laptop number 2. In the solarize UARK campus design problem, we obtained design data from 45 designers. The participants are indexed by their corresponding flash drive number in which the design details are provided. At the end of the design, participants return their flash drive and the corresponding design data are retrieved.

9.3.2 Data preparation and formulating the reinforcement learning model

We define the RL components, i.e., states, actions, rewards, in the context of the design problem as below:

States: State is the current situation in which the agent interacts with the environment. In this study, since our goal is to mimic human design behaviours, we define the states in RL as the state of the designers thought process during design. Various ontological models have been proposed to represent design processes and interpret design thinking [174]. In this study, the proposed state representation model is inspired by the function-behaviour-structure (FBS) design process model. We referred to the FBS model because it is a design ontology that can be used to represent a general system design problem. However, the original FBS model is not able to fully describe the design process in CAD environment. Therefore, the FBS model was later extended in the CAD context [175] where a major inclusion is to add the interpretation sub-process. So, we added interpretation in our model of representing the design thinking states. In total, we defined six design thinking states in CAD that include Formulation, Reformulation, Synthesis, Interpretation, Evaluation, and Analysis. These states are considered as the states for RL. From the collected design data, we observe that designers can perform the inter-state movement. For example, suppose a designer is in a state where they add a wall. After that, they move to another state where they have various options to perform such as editing the wall, analysing it, assessing the cost, or removing it. Using these options, designers can move from one state to any other state, which produces a fully connected network (Figure 9.3).

Actions: The actions in our RL problem are the design actions performed by designers. We observe that designers can perform similar categories of actions in each state. First, categories allow capturing the context-independent essence of design actions and provide better generalizability. Second, these categories significantly decrease the number of possible state-action pairs and reduce the computational burden during the training of RL agents. Finally, in our previous human-computer interaction experiments [176], where we train a deep learning model to recommend design actions to designers, we have found that designers feel interrupted if they were provided with a detailed suite of actions. Therefore, grouping design actions into a small number of categories provide a more condensed set of design recommendations in a potential extension of this study to human-computer collaboration. In this study, we identify six unique categories of design actions in the experimental data. These categories and the corresponding actions are listed in Table 2.

Reward: The reward is the feedback from the environment. The RL agent aims to maximize the total reward that is calculated by summing all the immediate rewards. However, this sum can potentially grow indefinitely. Therefore, a discount factor (γ) is included in the reward function to reduce the contribution of the future rewards. The reward can be expressed as follows:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ... \tag{9.3}$$

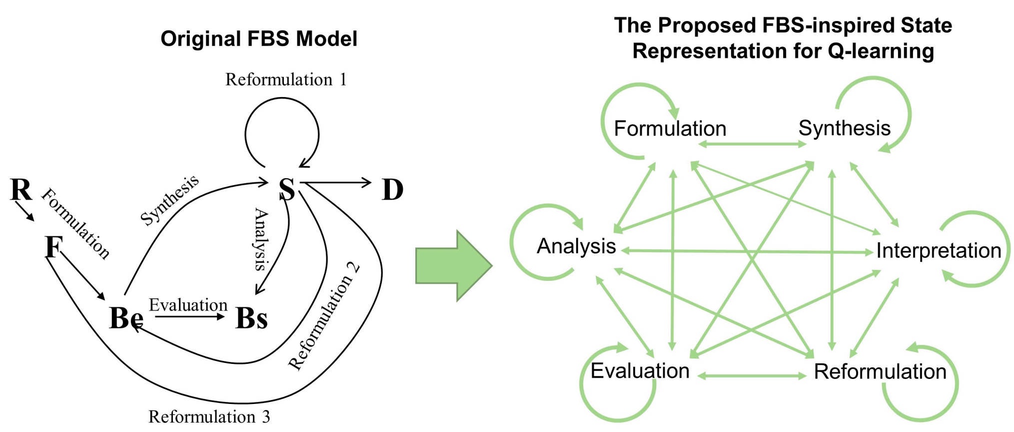


Figure 9.3: Figure 1. The FBS design process model (Kannengiesser et al., 2009) and the design thinking states defined in the proposed reinforcement learning model

Traditional RL is a self-learning method that uses a reward from the environment. As our target is to build an agent that mimics human designers, we use the data containing designers' actions in the solar system design experiment (e.g., those shown in Table 2) to generate a reward table. Combining this data-driven reward with the self-learning capability of RL is a unique aspect of this study. Figure 9.3 shows our overall approach to training an RL agent to mimic a human designer. We employ the first-order Markov chain model on the designers sequence to construct the reward table. From the first-order Markov chain, we obtain the transition probability matrix for each designer. Then, we average all the transition probability matrices from all designers to obtain a final reward table. This Markov chain-based reward mechanism reinforces most frequently appearing action pairs during training.

9.3.3 Model setup and evaluation

Baseline: We choose the first-order Markov chain model as our baseline model to compare with the Q-learning agent. From the first-order Markov chain, we obtain a transition probability matrix [36] for each designer in the dataset. We aggregate transition probability matrices from (n-1) designers and predict on the n^th designer's sequence. By iterating this process, we obtain the prediction accuracy for all designers in the data and use the average as the final prediction accuracy. The average prediction accuracy is about 41%.

Cross-validation: We use the k-fold cross-validation technique to evaluate the Q-learning agent. In this method, we split the dataset into k partitions. Then, k rounds of training and testing are performed in a way such that in each iteration, (k-1) partitions are used to obtain the reward table and train the Q-learning agent. The rest of the partition is used to test the Q-learning agent. In this study, 6 designers are used as test data and the rest of them as training. This process iterates through the entire dataset. In this way, we perform a total of 11 rounds of training and

Table 9.2: Design action categories and their corresponding actions

Design action category	Action Category	Example
Addition of any components	Add	Add wall, Add solar panel, etc.
Edit of any components	Edit	Edit door, Edit wall, etc.
Environmental check	Show	Show Helidon, Show sun path, etc.
Evaluation of cost	Cost	Cost
Removal of any components	Remove	Remove window, Remove the roof, etc.
Analysis of annual net energy	Analysis	Energy Annual Analysis

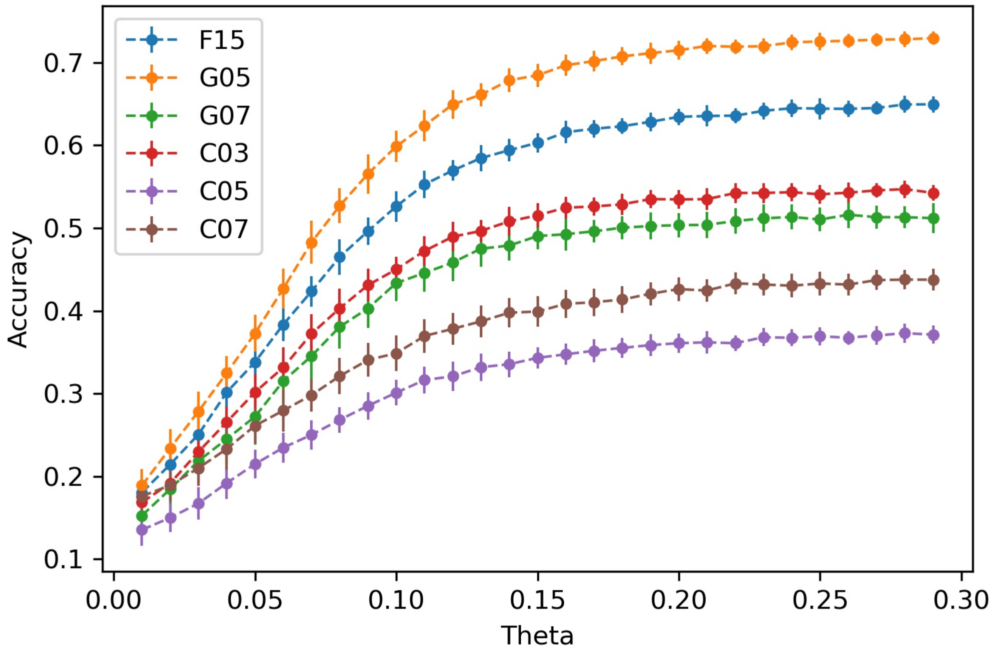


Figure 9.4: Prediction accuracy of the high-performing design group (left) and the low-performing design group (right)

testing.

Parameter settings: We determine the optimal settings for the hyperparameters of the RL model using trial and error and train the Q-learning agent based on those settings. We choose the learning rate (α) of this study as 0.3 and the discount factor (γ) as 0.6 for updating the Q-value (see Equation (1)). We update the Q-table by 10,000 iterations. To obtain the next possible action from the Q-table with the best accuracy, we also tune the value of θ. Note that the optimal settings for the model parameters will be different in other applications and should be tuned again based on the input data.

Metric for prediction accuracy: We compare the agents generated sequence with the actual sequence to evaluate the Q-learning agent. The agent generates the next sequence based on the previous action. As the agent chooses the final design action from a probability distribution, the prediction can vary from iteration to iteration. Therefore, to account for the stochasticity, we run a total of 50 realizations to generate each sequence. In each sequence, we compare the predicted actions with the actual decisions in the data, count the number of correctly predicted actions, and divide them by the total length of the sequence.Finally, we take the average of the 50 prediction accuracy. In this way, we obtain the final prediction accuracy using the following equation

$$Prediction accuracy = \frac{1}{50}\sum_{i=1}^{50}\left(\frac{n_i^{cp}}{L}\right) \tag{9.4}$$

where n^cp is the correctly predicted actions and L is the length of the designer's sequence.

9.4 Result and Discussion

We analyse the results from two different aspects. First, we discuss the result obtained from the Q-learning trained by the average reward of energy-plus dataset. Additionally, we compare it with Q-learning trained from individual reward through conducing design of experiments of all combination. Next, the trained Q-learning model from energy-plus dataset is tested on the solarize UARK campus dataset.

9.4.1 Results on energy-plus dataset

Result

Results obtained from different folds indicate that the prediction accuracy improves with the increase in θ, as shown in Figure 9.4. Initially, when θ is close to zero, the RL agent provides uniform distribution to select the next action at a given state, resulting in a random search. However, with the increase in θ, the probability of the reinforced action pairs identified in the data increases. Therefore, the accuracy of predicting the next design action also increases compared to a random search. Figure 9.4 shows a sample for the prediction accuracy from one of the folds out of 11 for the individuals F15, G05, G07, C03, C05, and C07. The prediction accuracy increases from $\theta = 0$ to 0.25. After that, the prediction accuracy does not increase significantly and saturates to its final value for all the design sequences tested. Among all the designers, G05 achieves the highest prediction accuracy of 73%, higher than the baseline prediction accuracy of 41% achieved by the Markov chain model. We also observe that in several folds, the prediction accuracy obtained from the maximum theta for a few designers is lower than the baseline accuracy. These results indicate that the agent reinforces specific design patterns (i.e., Edit-Edit, Edit-Analysis, etc.) and provides better accuracy for those designers who behaved such patterns but yield lower accuracy for those who did not.

The k-fold cross-validation results inspire us to investigate further the prediction accuracy based on the designers performance. We define design performance in the following equation:

$$Designperformance = \frac{ANE \times Budget}{Cost} \tag{9.5}$$

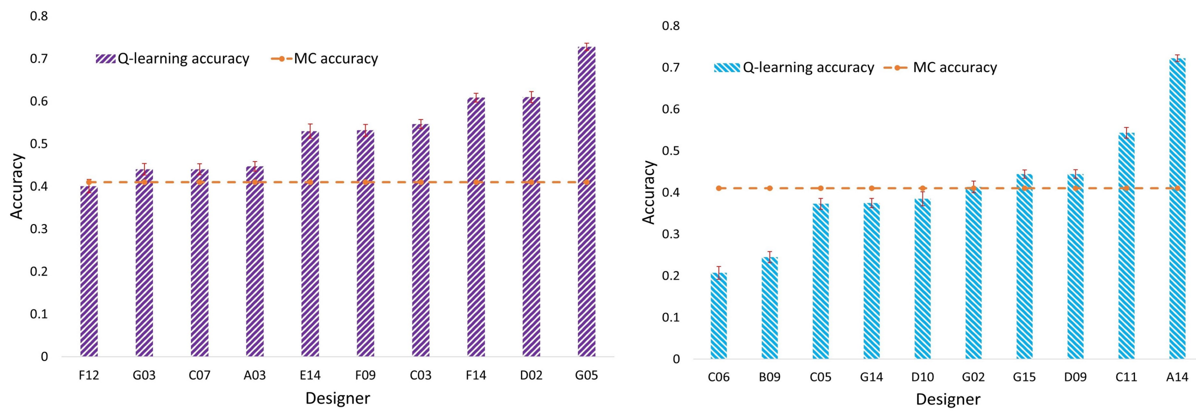


Figure 9.5: Prediction accuracy of the high-performing design group (left) and the low-performing design group (right)

We pick ten highest and lowest-performing designers and compare their prediction accuracy. We identify all the designers prediction accuracy in each group by training the rest of the designers. This means when considering the highest performing 10 designers, we train the reinforcement agent on the rest of the 42 designers and test on these 10 designers. We follow the same procedure for the lowest-performing 10 designers.

Figure 9.5 shows the prediction accuracy for the high-performance group. In this group, nine designers out of ten achieve high prediction accuracy ($> 41\%$) compared to the Markov chain model. While in the low-performance group, only four designers prediction accuracy is higher than the Markov chain prediction accuracy. It is worth mentioning that the highest performer among all the designers, G05, also achieves the highest prediction accuracy. To understand the design strategy, we compare the transition probability matrix of the highest (G05) and lowest performer (F12) in the high-performance group. Figure 9.6 shows the heatmap of the transition probability matrix. Bigger circles indicate a higher transition probability while smaller circles indicate a lower transition probability.

Discussion

The highest performer G05 follows only specific design patterns and is very focused on a few particular actions while the lowest performer F12 in this group uses a variety of design patterns. For example, the highest three transition probabilities for G05 are "Edit-Edit" (0.82), "Edit-Add" (0.71), "Edit-Analysis" (0.58). The transition probabilities of these patterns for F12 are 0.57, 0.46, and 0.25 respectively. Additionally, G05 did not use Analysis-Analysis, Cost-Remove, or Remove-Show action pairs at all during the process. However, F12 had these patterns during the design

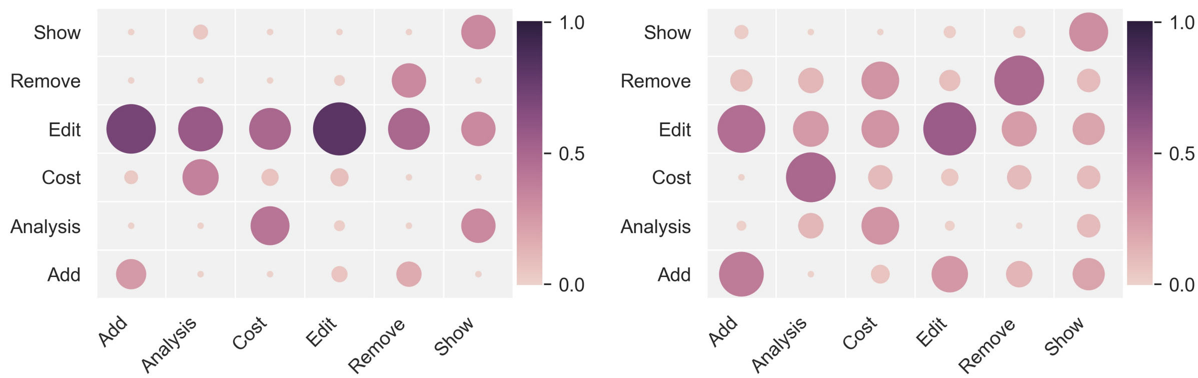


Figure 9.6: Prediction accuracy of the high-performing design group (left) and the low-performing design group (right)

process. Although both designers achieve good design performance as they were both in the good-performing group, G05 finished the design task by exploiting a few specific design patterns while F12 explored different patterns to reach the objective. The RL agent learns a particular set of consistent action pairs during training due to the reinforcement of frequently occurring behaviours, thus the prediction accuracy increases if the designers follow such a consistent design behaviour pattern.

In the low-performance group, most of the designers also use a variety of design patterns to explore the design space. The individual A14 achieves the highest prediction accuracy in this group. Similar to the highest prediction accuracy (G05) achieved in the high-performance group, A14 also achieve the highest performance in the low-performance group. Figure 6 shows the transition probability matrix of A14. Similar to G05, A14 also uses specific design patterns during the design process, but the design performance achieved by A14 is lower than G05. This may be attributed to the fact that A14 uses several redundant design action pairs. For example, the transition probability of using Analysis-Analysis for this designer is high but not necessary. This is because once Analysis (the analysis of ANE) is conducted, there is no need to perform the analysis again in the next action.

Additionally, this designer did not use Show at all which indicates that the designer is not interacting with the CAD environment and is not active in learning the solar science concepts underpinning the design problem. This result indicates that though high-performing and low-performing designers may have similar prediction accuracy, their design strategies could be different albeit consistent throughout the process. Here consistency refers to the behavioural pattern those designers use the same strategies (e.g., a pair of actions) over and over. For example, designers frequently use "Formulation → Edit" and "Edit → Edit", and they follow these strategies consistently even if they may not be improved design objectives. This is also congruent with the findings in the literature. [177] showed that both experts and consistently wrong non-experts can present such behaviours. But it is worth noting that our models do not rely only on consistency but also the previous design action data to predict future actions.

We compare the average Q-learning and Markov chain model with the individual Q-learning and Markov chain model accuracy. We carry out this comparison by conducting paired t-test using all the combinations of three aspects. Particularly, we compare the highest and lowest ten performer based on the reward/transition probability (i.e., average and individual) and types of agents (i.e., Q-learning and Markov chain). Table 9.3 shows all the combinations and their corresponding p-value. In the first column of the table, keeping the high performance and low performance design comparison constant, we interchangeably use the other variables (i.e., rewards and agents). To evaluate the statistical significance of the comparison by paired t-test, we set the corresponding null and alternative hypothesis. For example, to compare the prediction accuracy of Q-learning

Table 9.3: Statistical t-test between the prediction accuracy of different agents. High and low indicates high performance and low performance group. Avg and Ind indicates average reward and individual reward. Q and MC indicates Q-learning agent and Markov chain agent (i.e., High Q Avg indicates the the prediction accuracy of average Q-learning from high performance group

High performance **Low performance**	**Average reward** **Individual reward**	**Markov chain agent** **Q-learning agent**
High Q Avg Low Q Avg **p-value: 0.030**	High Q Avg High Q Ind **p-value: 0.025**	High MC Avg High Q Avg **P-value: 5.09E-05**
High MC Avg Low MC Avg p-value: 0.020	High MC Avg High MC Ind p-value: 0.49	Low MC \Avg Low Q Avg p-value: 0.19
High Q Ind Low Q Ind p-value: 0.25	Low Q Avg Low Q Ind p-value: 0.29	High MC Ind High Q Ind p-value: 0.09
High MC Ind Low MC Ind **p-value: 0.013**	Low MC Avg Low MC Ind **p-value: 0.0018**	Low MC Ind Low Q Ind **p-value: 0.017**

agent from the average reward of the high performance group and the Q-learning agent from the average reward of the low performance group, the null hypothesis (H_0) is that the their accuracy is equal; and the alternative hypothesis (H_a) is that former is significantly higher than the later. With the level of significance of 0.05, the p-value in the table indicates the Q-learning agent trained from the average reward of the high performance group achieved significantly higher accuracy than the others. Similarly, for the second and the third columns, we keep rewards and agents constant respectively while changing the other variables.

The shaded cells in the table 9.3 indicate the statistically significant comparisons. For example, the accuracy of the average Q- learning for high performance group is higher than the low performance group. The Q-learning reinforces the average reward for both groups. However, designers in the high-performance group use consistent good strategies and those strategies are further reinforced aggregately. Therefore, their accuracy is significantly higher than the low performance group. The accuracy of the average Q-learning is also significantly higher than individual Q learning in high performance group. In the average reward mechanism, all the prominent design strategies are reinforced while in the individual reward those strategies might be use in a low frequency. Therefore, for the high-performance group the accuracy of Q-average is significantly higher than the Q-individual. Similarly, the accuracy of the average Q-learning of the high performance group is significantly higher than the accuracy of the average Markov chain of the high performance group. Here the Q- learning reinforces the prominent design strategies. Therefore, Q-learning with average reward achieves significantly higher accuracy than the average Markov chain.

The individual Markov chain for low performance group achieved the significantly higher accuracy than the others. For example, it achieves significantly higher accuracy than the individual Markov chain from high performance group. As discussed earlier, in the low performance group, different designers use different design actions throughout their design task. Additionally, these design actions are equally used in different phases in design process. Therefore, prediction accuracy for the last portion of their design actions is higher. In contrast, high performance designers use specific design patterns in particular phase of the design. As a result, when calculating individual accuracy, Markov chain agent trained on their first portions of the design actions could not produce higher accuracy for the test dataset. The individual Markov chain for low performance group also achieved significantly higher accuracy than the average Markov chain for low performance group. This is because, averaging the different transition probabilities of low performance designers reduces the probability of choosing any particular design actions. Finally, the accuracy of the individual Q-learning for low performance group is significantly lower than the individual Markov chain of low performance group. Q-learning reinforces the design actions.

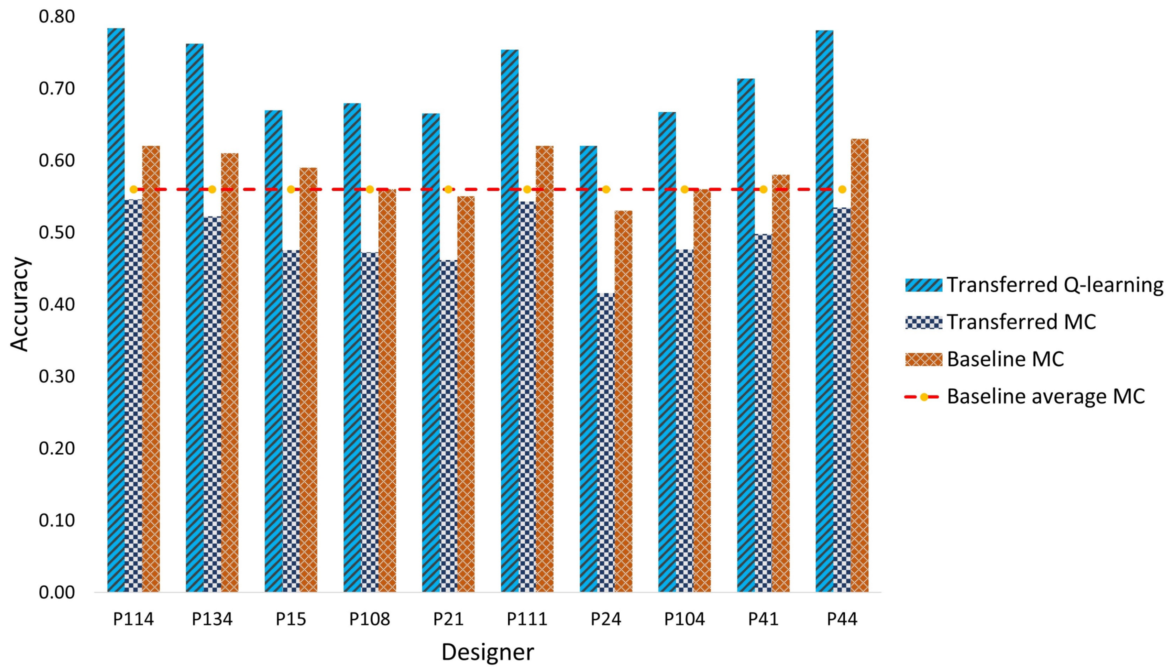


Figure 9.7: The prediction accuracy of the transferred Q-learning, Markov chain and the baseline Markov chain model for the high performance design

9.4.2 Results on design knowledge transfer

After training the Q-learning agent with the energy-plus dataset, we test the agent on solarize UARK campus dataset. Additionally, we test the transferablity of Markov chain model on this dataset. Particularly, We transfer the learned average Q-table for Q-learning agent and aggregated transition probability matrix for Markov chain model. Both of the models are compared with the baseline Markov chain model trained from the solarize UARK campus dataset. Like previous, we pick ten highest and lowest performance designers and compare their accuracy across different models. Using the transferred Q-learning and MC agent, we calculate the prediction accuracy for each of the ten highest and lowest performance designers. While for baseline MC model, we train it using the all the designers except the highest or lowest ten designers. Figure 9.7 shows the transferred Q-learning and Markov chain model accuracy along with the baseline Markov chain model for high performance designers. It shows that Q-learning achieves the highest accuracy for all the designers than the other agents. The highest prediction accuracy is achieved by the designer 114 and 44 which is 0.78. However, the prediction accuracy of the transferred Markov chain model is lower than the baseline model for all the student. The average Markov chain accuracy from the aggregated transition probability is found by 0.56.

For the low performance designers, the transferred Q-learning agent also achieved the highest prediction accuracy compared to the transferred Markov chain model and baseline Markov chain model (see Figure 9.8). However, the overall prediction accuracy of the transferred Q-learning is lower in the low performance group than the high performance group. Like energy-plus home design problem, in this problem, high performer designers also follow a specific design patterns. Q-learning agent is able to reinforce those patterns and therefore, the prediction accuracy is higher in high performance group. For some of the designers, the accuracy of the aggregated Markov chain is higher than the transferred the Q-learning accuracy. Like high performance design, the prediction accuracy for the transferred Markov chain is also lower than the baseline Markov chain model.

Figure 9.9 shows the correlation between transferred Q-learning and baseline Markov chain accuracy for high performance group and low performance group. It shows that there is a strong correlation between transferred Q learning and baseline Markov chain model for both group. Particularly the correlation co-efficient for high performance group is 0.94 and low performance group is 0.95. This high correlation indicates that the prediction accuracy for the transferred Q-learning is not redundant. As mentioned before, the Q-learning reinforces the most prominent design actions. Therefore, with the increase of the accuracy of baseline Markov chain, the accuracy of the transferred Q-learning also increases.

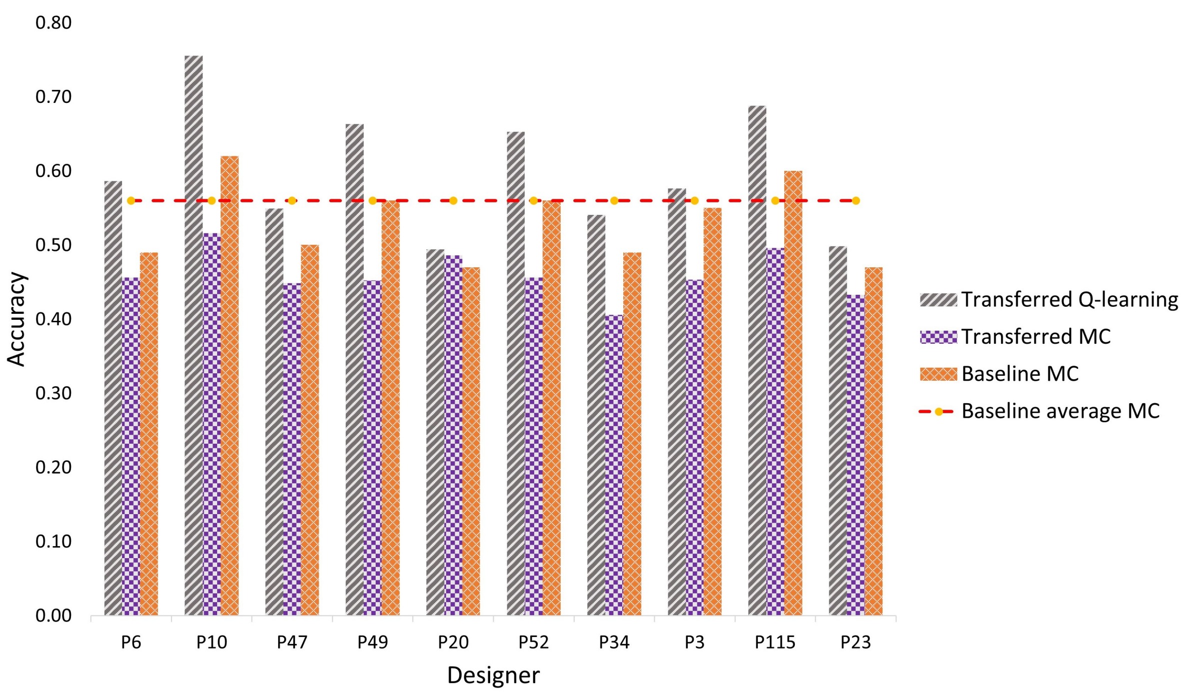


Figure 9.8: The prediction accuracy of the transferred Q-learning, Markov chain and the baseline Markov chain model for the low performance design

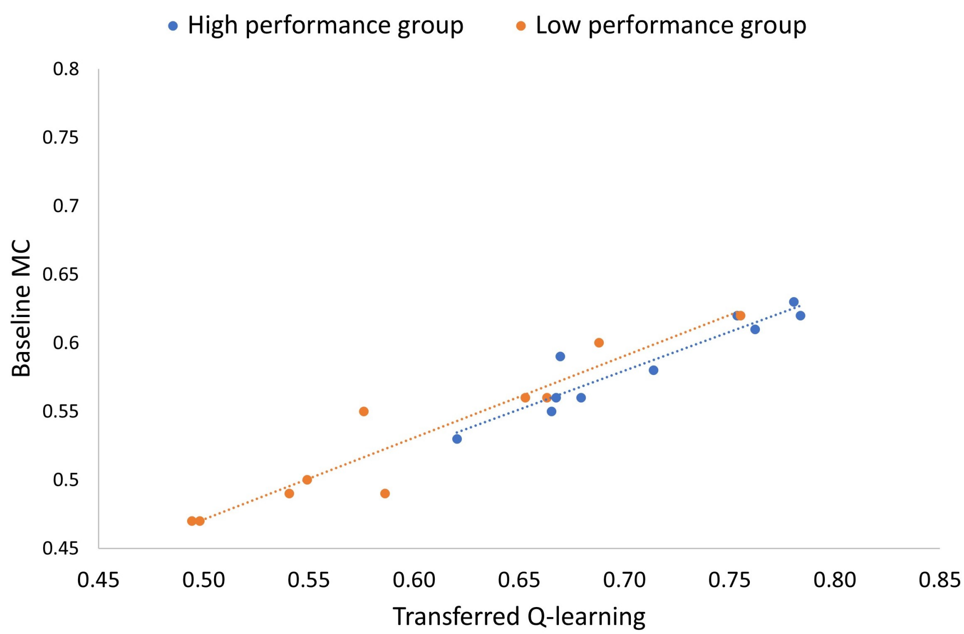


Figure 9.9: The correlation between transferred Q learning and baseline Markov chain accuracy for high performance group(left) and low performance group (right)

9.5 Conclusion

In this study, a design agent is developed based on Q-learning, which is a commonly used method in the reinforcement learning literature, to mimic human design strategies. The overall contribution of this study is threefold. First, the model trained using a reinforcement learning algorithm is novel in human behaviour exploration. We train this model by leveraging a data-driven reward mechanism based on the first-order Markov chain model. Once the model is trained, it chooses design actions from a probability function controlled by a specific model parameter, θ. To the best of our knowledge, this is the first study using reinforcement learning to understand human design strategies. Second, this model provides several important design behaviours. For example, increasing the model parameter θ also increases the prediction accuracy. Additionally, in most of the cases, compared to the Markov chain baseline model, the agent provides better prediction accuracy for high-performing designers. We also observe that certain strategies-patterns differentiate some of the high-performing designers from low-performing designers. The design agent learns those design patterns and provides higher prediction accuracy for most high-performing designers. Third, the model successfully transfers the design knowledge from one design problem to another design problem. We identify that with the transferred knowledge, the Q-learning agent perform better than the transferred Markov chain model and baseline Markov chain model. There are some limitations in the current study. First, this approach does not evaluate design strategies, but rather predicts future design actions and can identify beneficial design patterns to improve the design objective. Second, although we train the agent with average design behaviour data, the prediction accuracy is higher only for high-performance design groups when these designers follow a consistent design pattern.

10 Conclusion and Future Work

This dissertation developed and demonstrated a systematic data-driven research framework for design thinking research, including developing a computer-aided design (CAD) based research platform, conducting design experiments, performing data collection, modeling, analyzing, and predicting human sequential decision-making in complex systems design. The overall contributions of this dissertation can be summarized as follows.

- This dissertation creates a new avenue for design thinking research in the context of systems design through a newly developed research platform in a CAD environment.
- This dissertation presents novel techniques which leverage a Markov chain model and different clustering algorithms for clustering designers' one-step sequential behaviors.
- Based on existing design theories, this dissertation research identified five design behaviors that include design action preference, one-step sequential behavior, contextual behavior, long-step sequential behavior, and reflective thinking. The latent representations of these behaviors obtained from deep neural network models are referred to as Design Embeddings. The embeddings can be used to identify and group designers with similar behavioral patterns that can be helpful in forming efficient and effective design teams. Another benefit of design embeddings is that they can be used to identify useful design patterns and strategies that are beneficial to train novice designers.
- In this dissertation, two different predictive models are developed for the prediction of sequential design decisions. By integrating the design process model into various machine learning techniques, the resulting deep-learning models are found to outperform the models traditionally used to predict sequential design decisions in design research. These models are general and can be applied to model designers' sequential design sequence in any design problem as long as their sequential design action data are available.
- This dissertation corroborates the transfer of design knowledge using a reinforcement learning-based design agent.

In summary, this dissertation creates a stepping stone towards an in-depth understanding of designers' sequential design decision-making and better approaches to predicting sequential design decisions, and therefore supports human-AI collaboration of the future.

We revisit the research questions we aim to answer at the beginning of the dissertation.

- RQ1: What are the relationships between design behaviors and design outcomes?
- RQ2: How to predict designers' sequential decisions in computer-aided design based on the characterization of different aspects of design thinking?

To answer RQ1, we first develop an open-source research platform for collecting design data. To model designers' one-step sequential behavior, we leverage the Markov chain model in Chapter 4. In Chapter 5, we then develop approaches that model design behaviors from multiple dimensions, which includes action behavior, contextual behavior, long-step sequential behavior, reflective thinking behavior in addition to the one-step sequential behavior. From these approaches, we identified beneficial design patterns such as "Synthesis $\longrightarrow$ Synthesis" and "Reformulation $\longrightarrow$ Formulation" that are commonly used in CAD environment. In addition, we clustered the embedding and identified designers with similar behavioral patterns. Then, designers' performance in terms of the qualities of their designs (e.g., the annual net energy of the solarized home design and the cost) is compared among different clusters. The results indicate that even if the sequential design strategies of designers could be different, their final design qualities could be similar. This is because the design process is typically a result and reflection of a combination of different design behaviors. Instead of a single behavior, multiple different types of design behavior work together to influence the design process and, therefore, the design outcome. In the study on design transferability, we also observed that high-performing designers are very focused on a few specific design patterns that can help exploit the design space, such as the "Edit $\longrightarrow$ Add" sequence, while most of the low-performing designers' sequential design actions are lack of prominent patterns. There were a few low-performing designers who used specific design patterns, such as "Analysis $\longrightarrow$ Analysis," but since those patterns were redundant, they were essentially useless for improving design performance.

To answer RQ2, we developed several predictive models by integrating design process models and designers' attributes, both of which provide additional design thinking features that can potentially improve the models' predictive performance. First, in Chapter 6, we developed a framework by integrating the FBS design process model to predict future design actions. The model successfully predicted future design decisions. The prediction accuracy (61.25%) of this model is higher than the baseline random model (14.15%) and the repetitive model (51.2%). We then extend this framework in Chapter 8 by two methods (i.e., direct method and indirect method) that combine designers' static information (i.e., designers' attribute that does not change over time) into their dynamic design action sequence to further improve the models' predictive performance. The combination of static and dynamic data significantly improves the prediction accuracy. In particular, the direct method achieved 63.38% prediction accuracy, while the indirect method achieved 61.98%. The methods are compared with a baseline model that only takes dynamic data into RNN. The prediction accuracy of that baseline model is 60.91 %. The result indicates that the integration of static data improves the prediction of sequential design decisions. Finally, in Chapter 9, a reinforcement learning-based design agent was developed to mimic human design behaviors. The design agent is then used to test the transferrability of design knowledge learned from source design problems to target design problems.

Several future directions are identified on the basis of this dissertation. They are described below.

- **Improving prediction accuracy using transformer model**

  To predict future design actions based on previous actions, we used the LSTM model. LSTM is able to capture long-term sequence; therefore, the prediction accuracy is higher than that of the other traditionally used models. However, in the LSTM model, the design action data is input into the LSTM model sequentially, and parallelism is not possible. Therefore, the problem of vanishing/exploding gradients still exists in the LSTM model. As a result, for a very long design sequence (i.e., 700 design actions), LSTM will not work very well. Recently, transformer models were found to outperform the LSTM model in sequential tasks [178]. The transformer model is an encoder-decoder model. The encoder extracts features from the input sequence, and the decoder uses those features to produce an output sequence. For predicting the next design action, either the encoder part or the decoder part with a final prediction layer can be used. The transformer model allows parallel computing and thus can reduce training time.

- **Adding fairness/inclusiveness into the machine learning models**

  Fairness in machine learning is an active area of research. The goal of fairness in machine learning is to understand and prevent bias in data and models related to race, income, demographics, culture, and other characteristics historically associated with discrimination. Design data also includes designers of different cultures, demographics, and backgrounds. The data may contain a different number of designers of different demographics and cultures. For this reason, deep learning predictive models trained from unbalanced data could produce bias predictions. To solve the biased data distribution, during data collection, it would be the best to ensure that each group contains a sufficient and equal size of data set. Additionally, after developing and training the model with the collected data, it should be evaluated for fairness. If the model is used for prediction purposes or as a recommendation system for designers, the impact of different types of error (e.g., false positive and false negative) produced by the model for different user groups might have an adverse effect on their designs. Therefore, for each group, the way the model predicts, in particular what the prediction accuracy is for each of the groups, should be evaluated [170].

- **Adding psychological attributes to deep learning model:**

  From our previous study, it is observed that integrating static features such as design attributes improves prediction accuracy. Design is a cognitive process, and different psychological factors (i.e., convergent thinking, divergent thinking, working memory, etc.) may influence how designers make design decisions. Therefore, including these factors in the predition models could increase the models' predictive performance. For example, convergent thinking and divergent thinking are studied by cognitive processes that are related to creative performance. Divergent thinking generally refers to the cognitive processes related to idea generation and is traditionally measured through the novelty (also referred to as originality) and the number (fluency) of generated ideas. Convergent thinking instead refers to the processes that allow one to choose the correct solution to a problem out of a pool of potential solutions. We can measure creative cognition through the originality/novelty and fluency of divergent thinking. The identified cognitive measures can be added as static data in the deep learning models that we developed in Chapter 8.

- **Use embedding to enable action level prediction**

  In our study, we map design actions to the design process stage and predict future design decisions based on previous historical data. Although this process model generalizes design as a context-independent problem and helps to understand design thinking at the higher level, another lower-level model can be developed for design action prediction. In this model, an embedding layer can be added after the input design sequence. The embedding technique identifies the relationship among the design actions and at the same time reduces the dimensionality. Embedding can be obtained by using several techniques, such as Word2Vec and the bidirectional LSTM auto-encoder. The embedding layer can then be used after the input layer. In this way, we can directly predict design actions.

- **Training design agent by segregating design action sequences**

  In this dissertation, a reinforcement learning learning-based agent has been developed to mimic human design behavior. In particular, we use a Markov chain-based data-driven reward mechanism that considers the full-length design sequence. However, design may have several design phases, and in each phase, the importance of different design actions may be different. Therefore, instead of extracting a reward function from the entire design sequence, using the reward functions extracted from each design phase may help improve the

Q-learning model's performance in predicting the design actions in a particular design stage. Additionally, in this dissertation, for transfer learning, we used design problems in the same application domain, i.e., the solar system design, but with different complexity and design variables. In future studies, design problems from different application domains, other than solar design problem, can be chosen to test the transferability of design knowledge.

- **From reinforcement learning to imitation learning**

  In order to test the transferrability of design knowledge, in this dissertation, we developed a reinforcement learning model using a data-driven reward formulation. In particular, we use an offline reinforcement learning technique that tries to maximize a data-driven reward function based on the first-order Markov chain of design sequences. Such a data-driven reward formulation ensures that the model tries to mimic human behaviors and reinforces the most frequent (maybe beneficial) one-step sequential behaviors. In addition to reinforcement learning, imitation learning can also be used in this study. Imitation Learning (IL) is a training method in which the agent imitates human behavior. In IL, instead of using a reward function, an expert, usually a human, provides the agent with a set of demonstrations. The agent then tries to learn the optimal policy by following and mimicking the experts decisions. Like reinforcement learning, the main component of IL is the environment which is essentially a Markov decision process (MDP). The environment has a set of states (S) and a set of actions (A) and unknown rewards (R). The agent performs different actions according to the expert's policy (π). The policy in IL is typically learned using a supervised learning method and updated by a loss function. We can use deep learning to obtain the policy and use them in IL to predict design actions.

- **Theoretical advancement on design process model**

  In this dissertation, only the function-behavior-structure (FBS) design process model was used to encode the design actions collected from the design problems in Energy3D. Although the FBS design process is a widely accepted model and can be potentially used in any design problem regardless of the design context, we used it in the CAD environment and mainly for solar design problems. However, in other real-world design problems, such as Design for the Environment (DfE), there may be additional design process that may not be fully encoded by the FBS design process model. For example, in DfE, life cycle assesment is a primary process that includes several actions such as raw metarial extraction, manufacturing, distribution, use, disposal, etc. for analyzing the potential environmental impact on designed products. However, the FBS design process model does not have any design process stage that maps such design actions. In that case, we can modify and extend the existing models to develop a new model. The encoded data from this new design process model can then be used for further analyses and prediction tasks.

Bibliography

[1] J. S. Gero, "Design prototypes: a knowledge representation schema for design," *AI magazine*, vol. 11, no. 4, p. 26, 1990.

[2] M. Rahman, C. Schimpf, C. Xie, and Z. Sha, "A cad-based research platform for data-driven design thinking studies," *Journal of Mechanical Design*, pp. 1–44, 2019.

[3] D. Braha and O. Maimon, "The design process: properties, paradigms, and structure," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 27, no. 2, pp. 146–166, mar 1997.

[4] J. Mostow, "Toward Better Models of the Design Process," *AI Magazine*, vol. 6, no. 1 SE - Articles, p. 44, mar 1985. [Online]. Available: https://ojs.aaai.org/index.php/aimagazine/article/view/468

[5] N. Cross and R. Roy, *Engineering design methods*. Wiley New York, 1989, vol. 4.

[6] Y. ISHINO and Y. A. N. JIN, "Acquiring engineering knowledge from design processes," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 16, no. 2, pp. 73–91, 2002. [Online]. Available: https://www.cambridge.org/core/article/acquiring-engineering-knowledge-from-design-processes/FA997A9B750F4AD154FF4D7534D4302A

[7] J. W. Herrmann, "Progressive Design Processes and Bounded Rational Designers," *Journal of Mechanical Design*, vol. 132, no. 8, 07 2010, 081005. [Online]. Available: https://doi.org/10.1115/1.4001902

[8] C. L. Dym, A. M. Agogino, O. Eris, D. D. Frey, and L. J. Leifer, "Engineering design thinking, teaching, and learning," *Journal of Engineering Education*, vol. 94, no. 1, pp. 103–120, 2005.

[9] D. Crismond and R. Adams, "The Informed Design Teaching and Learning Matrix," *Journal of Engineering Education*, vol. 101, pp. 738–797, 2012.

[10] M. Dinar, J. J. Shah, J. Cagan, L. Leifer, J. Linsey, S. M. Smith, and N. V. Hernandez, "Empirical Studies of Designer Thinking: Past, Present, and Future," *Journal of Mechanical Design*, vol. 137, no. 2, feb 2015. [Online]. Available: https://doi.org/10.1115/1.4029025

[11] M. T. Greene, R. Gonzalez, and P. Y. Papalambros, "Measuring systems engineering and design thinking attitudes," in *Proceedings of the Design Society: International Conference on Engineering Design*, vol. 1, no. 1. Cambridge University Press, 2019, pp. 3939–3948.

[12] G. A. Hazelrigg, *Fundamentals of decision making for engineering design and systems engineering*, 2012.

[13] J. S. Gero and J. Milovanovic, "A framework for studying design thinking through measuring designers' minds, bodies and brains," *Design Science*, vol. 6, p. e19, 2020. [Online]. Available: https://www.cambridge.org/core/article/framework-for-studying-design-thinking-through-measuring-designers-minds-bodies-and-brains/04EC7F5384E423992E61CC8C65EA355F

[14] J. S. Gero and T. Mc Neill, "An approach to the analysis of design protocols," *Design Studies*, vol. 19, no. 1, pp. 21–61, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X9700015X

[15] C. Eastman, W. Newstetter, and M. McCracken, *Design knowing and learning: Cognition in design education*. Elsevier, 2001.

[16] C. A. Toh and S. R. Miller, "Choosing creativity: the role of individual risk and ambiguity aversion on creative concept selection in engineering design," *Research in Engineering Design*, vol. 27, no. 3, pp. 195–219, Jul. 2016. [Online]. Available: https://doi.org/10.1007/s00163-015-0212-1

[17] J. Gero, R. Yu, J. Wells *et al.*, "Creative design cognition differences between high school students with and without design education," in *DS 89: Proceedings of The Fifth International Conference on Design Creativity (ICDC 2018), University of Bath, Bath, UK*, 2018, pp. 240–247.

[18] D. P. Moreno, M. C. Yang *et al.*, "Creativity in transactional design problems: non-intuitive findings of an expert study using scamper," in *DS 77: Proceedings of the DESIGN 2014 13th International Design Conference*, 2014, pp. 569–578.

[19] M. L. Maher, L. Lee, J. S. Gero, R. Yu, and T. Clausner, "Characterizing tangible interaction during a creative combination task," in *Design Computing and Cognition '16*, J. S. Gero, Ed. Cham: Springer International Publishing, 2017, pp. 39–58.

[20] C. Toh, S. Miller, and G. E. O. Kremer, "The role of personality and team-based product dissection on fixation effects." *Advances in Engineering Education*, vol. 3, no. 4, p. n4, 2013.

[21] D. P. Moreno, L. T. Blessing, M. C. Yang, A. A. Hernández, and K. L. Wood, "Overcoming design fixation: Design by analogy studies and nonintuitive findings," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 30, no. 2, p. 185199, 2016.

[22] V. K. Viswanathan and J. S. Linsey, "Design fixation and its mitigation: A study on the role of expertise," *J. Mech. Des*, vol. 135, no. 5, 5/15. [Online]. Available: https://doi.org/10.1115/1.4024123

[23] C. A. Toh and S. R. Miller, “The impact of example modality and physical interactions on design creativity,” *J. Mech. Des*, vol. 136, no. 9, 5/15. [Online]. Available: https://doi.org/10.1115/1.4027639

[24] V. Viswanathan and J. Linsey, “Examining design fixation in engineering idea generation: the role of example modality,” *null*, vol. 1, no. 2, pp. 109–129, Apr. 2013. [Online]. Available: https://doi.org/10.1080/21650349.2013.774689

[25] J. S. Linsey, K. L. Wood, and A. B. Markman, “Modality and representation in analogy,” *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 22, no. 2, pp. 85–100, 2008. [Online]. Available: https://www.cambridge.org/core/article/modality-and-representation-in-analogy/F567723948DE2E845EDF1E048C87D9EF

[26] Q. Bao, D. Faas, and M. Yang, “Interplay of sketching & prototyping in early stage product design,” *null*, vol. 6, no. 3-4, pp. 146–168, Oct. 2018. [Online]. Available: https://doi.org/10.1080/21650349.2018.1429318

[27] Z. Bilda, J. S. Gero, and T. Purcell, “To sketch or not to sketch? that is the question,” *Design Studies*, vol. 27, no. 5, pp. 587–613, Sep. 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X06000317

[28] H. H. Tang, Y. Y. Lee, and J. S. Gero, “Comparing collaborative co-located and distributed design processes in digital and traditional sketching environments: Aăprotocol study using the function-behaviour-structure coding scheme,” *Design Studies*, vol. 32, no. 1, pp. 1–29, Jan. 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X10000475

[29] A. Menezes and B. Lawson, “How designers perceive sketches,” *Design Studies*, vol. 27, no. 5, pp. 571–585, Sep. 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X06000305

[30] M. Kavakli and J. S. Gero, “The structure of concurrent cognitive actions: a case study on novice and expert designers,” *Design Studies*, vol. 23, no. 1, pp. 25–40, Jan. 2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X01000217

[31] L. J. Ball, T. C. Ormerod, and N. J. Morley, “Spontaneous analogising in engineering design: a comparative analysis of experts and novices,” *Design Studies*, vol. 25, no. 5, pp. 495–508, Sep. 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X04000353

[32] S. Ahmed and B. T. Christensen, “An in situ study of analogical reasoning in novice and experienced design engineers,” *J. Mech. Des*, vol. 131, no. 11, 5/15. [Online]. Available: https://doi.org/10.1115/1.3184693

[33] J. W. Kan and J. S. Gero, “Using the fbs ontology to capture semantic design information in design protocol studies,” in *About: Designing. Analysing Design Meetings*. CRC Press, 2009, pp. 213–229.

[34] C. McComb, J. Cagan, and K. Kotovsky, “Mining process heuristics from designer action data via hidden Markov models,” *Journal of Mechanical Design*, vol. 139, no. 11, p. 111412, 2017.

[35] M. H. Rahman, C. Schimpf, C. Xie, and Z. Sha, “A Computer-Aided Design Based Research Platform for Design Thinking Studies,” *Journal of Mechanical Design*, vol. 141, no. 12, 2019.

[36] M. Rahman, M. Gashler, C. Xie, and Z. Sha, “Automatic clustering of sequential design behaviors,” in *Proceedings of the ASME Design Engineering Technical Conference*, vol. 1B-2018, 2018.

[37] M. H. Rahman, C. Xie, and Z. Sha, “Predicting Sequential Design Decisions Using the Function-Behavior-Structure Design Process Model and Recurrent Neural Networks,” *Journal of Mechanical Design*, pp. 1–46, jan 2021. [Online]. Available: https://doi.org/10.1115/1.4049971

[38] M. Suwa and B. Tversky, “What do architects and students perceive in their design sketches? A protocol analysis,” *Design Studies*, vol. 18, no. 4, pp. 385–403, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X97000082

[39] J. J. SHAH, N. O. E. VARGAS-HERNANDEZ, J. D. SUMMERS, and S. KULKARNI, “Collaborative Sketching (C-Sketch) An Idea Generation Technique for Engineering Design,” *The Journal of Creative Behavior*, vol. 35, no. 3, pp. 168–198, sep 2001. [Online]. Available: https://doi.org/10.1002/j.2162-6057.2001.tb01045.x

[40] S. Matthiesen, M. Meboldt, A. Ruckpaul, and M. Mussgnug, “Eye Tracking, a Method for Engineering Design Research on Engineers’ Behavior while Analyzing Technical Systems,” 2013.

[41] D. Pelleg and A. W. Moore, “X-means: Extending k-means with efficient estimation of the number of clusters.” in *Icml*, vol. 1, 2000, pp. 727–734.

[42] J. H. Panchal, Z. Sha, and K. N. Kannan, “Understanding design decisions under competition using games with information acquisition and a behavioral experiment,” *Journal of Mechanical Design*, vol. 139, no. 9, p. 091402, 2017.

[43] Z. Sha, K. N. Kannan, and J. H. Panchal, “Behavioral experimentation and game theory in engineering systems design,” *Journal of Mechanical Design*, vol. 137, no. 5, p. 051405, 2015.

[44] M. Shergadwala, K. N. Kannan, and J. H. Panchal, “Understanding the impact of expertise on design outcome: An approach based on concept inventories and item response theory,” no. V003T04A016, 5/15. [Online]. Available: https://doi.org/10.1115/DETC2016-59038

[45] C. A. Gosnell and S. R. Miller, "But is it creative? delineating the impact of expertise and concept ratings on creative concept selection," *J. Mech. Des*, vol. 138, no. 2, 5/15. [Online]. Available: https://doi.org/10.1115/1.4031904

[46] J. Sutera, M. C. Yang, and C. Elsen, "The impact of expertise on the capture of sketched intentions: perspectives for remote cooperative design," in *International Conference on Cooperative Design, Visualization and Engineering.* Springer, 2014, pp. 245–252.

[47] N. Cross, "Expertise in design: an overview," *Design Studies*, vol. 25, no. 5, pp. 427–441, Sep. 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X04000316

[48] C. A. Toh and S. R. Miller, "How engineering teams select design concepts: A view through the lens of creativity," *Design Studies*, vol. 38, pp. 111–138, May 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X15000174

[49] C. McComb, J. Cagan, and K. Kotovsky, "Rolling with the punches: An examination of team performance in a design task subject to drastic changes," *Design Studies*, vol. 36, pp. 99–121, Jan. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X14000659

[50] L. J. Ball and B. T. Christensen, "Analogical reasoning and mental simulation in design: two strategies linked to uncertainty resolution," *Design Studies*, vol. 30, no. 2, pp. 169–186, Mar. 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X08001294

[51] W. Visser, "Two functions of analogical reasoning in design: a cognitive-psychology approach," *Design Studies*, vol. 17, no. 4, pp. 417–434, Oct. 1996. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X96000208

[52] N. V. Hernandez, J. J. Shah, and S. M. Smith, "Understanding design ideation mechanisms through multilevel aligned empirical studies," *Design Studies*, vol. 31, no. 4, pp. 382–410, Jul. 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X1000027X

[53] Z. Sha, K. N. Kannan, and J. H. Panchal, "Behavioral experimentation and game theory in engineering systems design," *J. Mech. Des*, vol. 137, no. 5, 5/15. [Online]. Available: https://doi.org/10.1115/1.4029767

[54] P. T. Grogan and O. L. de Weck, "Collaboration and complexity: an experiment on the effect of multi-actor coupled design," *Research in Engineering Design*, vol. 27, no. 3, pp. 221–235, Jul. 2016. [Online]. Available: https://doi.org/10.1007/s00163-016-0214-7

[55] T. Alelyani, Y. Yang, and P. T. Grogan, "Understanding designers behavior in parameter design activities," no. V007T06A030, 5/15. [Online]. Available: https://doi.org/10.1115/DETC2017-68335

[56] H. Yao and M. Y. Ren, "Impressionist: a 3d peekaboo game for crowdsourcing shape saliency," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 50190. American Society of Mechanical Engineers, 2016, p. V007T06A025.

[57] P. T. Grogan and O. L. de Weck, "Interactive simulation games to assess federated satellite system concepts," in *2015 IEEE Aerospace Conference*, 7-14, pp. 1–13.

[58] J. A. Self, "Communication through design sketches: implications for stakeholder interpretation during concept design," *Design Studies*, vol. 63, pp. 1–36, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X19300043

[59] L. Sun, W. Xiang, C. Chai, Z. Yang, and K. Zhang, "Designers' perception during sketching: An examination of Creative Segment theory using eye movements," *Design Studies*, vol. 35, no. 6, pp. 593–613, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X14000428

[60] T. A. Nguyen and Y. Zeng, "A physiological study of relationship between designer's mental effort and mental stress during conceptual design," *Computer-Aided Design*, vol. 54, pp. 3–18, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010448513002005

[61] L. Liu, T. A. Nguyen, Y. Zeng, and A. B. Hamza, "Identification of Relationships Between Electroencephalography (EEG) Bands and Design Activities," aug 2016. [Online]. Available: https://doi.org/10.1115/DETC2016-59104

[62] K. Goucher-Lambert, J. Moss, and J. Cagan, "A neuroimaging investigation of design ideation with and without inspirational stimuliunderstanding the meaning of near and far stimuli," *Design Studies*, vol. 60, pp. 1–38, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X18300553

[63] T. Shealy and J. Gero, "The Neurocognition of Three Engineering Concept Generation Techniques," *Proceedings of the Design Society: International Conference on Engineering Design*, vol. 1, no. 1, pp. 1833–1842, 2019. [Online]. Available: https://www.cambridge.org/core/article/neurocognition-of-three-engineering-concept-generation-techniques/D44208044E9ED3072845A0D661FE1829

[64] S. Vieira, J. Gero, J. Delmoral, V. Gattol, C. Fernandes, M. Parente, and A. Fernandes, "Understanding the Design Neurocognition of Mechanical Engineers When Designing and Problem-Solving," aug 2019. [Online]. Available: https://doi.org/10.1115/DETC2019-97838

[65] S. Pachidi, M. Spruit, and I. van de Weerd, "Understanding users' behavior with software operation data mining," *Computers in Human Behavior*, vol. 30, pp. 583–594, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0747563213002884

[66] Y. Jin and Y. Ishino, "DAKA: design activity knowledge acquisition through data-mining," *International Journal of Production Research*, vol. 44, no. 14, pp. 2813–2837, jul 2006. [Online]. Available: https://doi.org/10.1080/00207540600654533

[67] J. Gopsill, C. Snider, L. Shi, and H. Ben, "Computer aided design user interaction as a sensor for monitoring engineers and the engineering design process," in *DS 84: Proceedings of the DESIGN 2016 14th International Design Conference*, 2016, pp. 1707–1718.

[68] C. Sen, Q. Parrish, and O. Galil, "Measuring Information Content of Freehand Sketches Using a Cognitive Chunk Visualization Protocol," in *ASME 2017 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2017, pp. V007T06A035–V007T06A035.

[69] C. McComb, J. Cagan, and K. Kotovsky, "Capturing human sequence-learning abilities in configuration design tasks through markov chains," *Journal of Mechanical Design*, vol. 139, no. 9, p. 91101, 2017.

[70] B. Y. Yu, T. Honda, M. Sharqawy, and M. Yang, "Human behavior and domain knowledge in parameter design of complex systems," *Design Studies*, vol. 45, pp. 242–267, Jul. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X16300230

[71] P. Egan, C. Schunn, J. Cagan, and P. LeDuc, "Improving human understanding and design of complex multi-level systems with animation and parametric relationship supports," *Design Science*, vol. 1, no. e3, p. e3, 2015. [Online]. Available: https://www.cambridge.org/core/article/improving-human-understanding-and-design-of-complex-multilevel-systems-with-animation-and-parametric-relationship-supports/94D2C5D1AEC11B5E4BBC46CD497B1DE1

[72] J. Gopsill, C. Snider, L. Shi, H. Ben *et al.*, "Computer aided design user interaction as a sensor for monitoring engineers and the engineering design process," in *DS 84: Proceedings of the DESIGN 2016 14th International Design Conference*, 2016, pp. 1707–1718.

[73] J. M. Ritchie, R. C. W. Sung, H. Rea, T. Lim, J. R. Corney, and I. Howley, "The use of non-intrusive user logging to capture engineering rationale, knowledge and intent during the product life cycle," in *PICMET '08 - 2008 Portland International Conference on Management of Engineering Technology*, 2008, pp. 981–989.

[74] Z. Sha, K. N. Kannan, and J. H. Panchal, "Behavioral experimentation and game theory in engineering systems design," *Journal of Mechanical Design*, vol. 137, no. 5, p. 51405, 2015.

[75] A. Sivanathan, T. Lim, J. Ritchie, R. Sung, Z. Kosmadoudi, and Y. Liu, "The application of ubiquitous multimodal synchronous data capture in CAD," *Computer-Aided Design*, vol. 59, pp. 176–191, 2015.

[76] R. C. W. Sung, J. R. Corney, and D. E. R. Clark, "Automatic Assembly Feature Recognition and Disassembly Sequence Generation," *Journal of Computing and Information Science in Engineering*, vol. 1, no. 4, pp. 291–299, 2001. [Online]. Available: http://dx.doi.org/10.1115/1.1429931

[77] R. Yu and J. S. Gero, "An empirical basis for the use of design patterns by architects in parametric design," *International Journal of Architectural Computing*, vol. 14, no. 3, pp. 289–302, May 2022. [Online]. Available: https://doi.org/10.1177/1478077116663351

[78] C. McComb, J. Cagan, and K. Kotovsky, "Capturing human sequence-learning abilities in configuration design tasks through markov chains," *Journal of Mechanical Design*, vol. 139, no. 9, p. 091101, 2017.

[79] ——, "Mining process heuristics from designer action data via hidden markov models," *Journal of Mechanical Design*, vol. 139, no. 11, p. 111412, 2017.

[80] T. Sexton and M. Y. Ren, "Learning an optimization algorithm through human design iterations," *Journal of Mechanical Design*, vol. 139, no. 10, p. 101404, 2017.

[81] A. M. Chaudhari and J. H. Panchal, "An experimental study of human decisions in sequential information acquisition in design: Impact of cost and task complexity," in *Research into Design for a Connected World*. Springer, 2019, vol. 134, pp. 321–332.

[82] A. M. Chaudhari, I. Bilionis, and J. H. Panchal, "Descriptive models of sequential decisions in engineering design: An experimental study," *Journal of Mechanical Design*, vol. 142, no. 8, 2020.

[83] M. Shergadwala, I. Bilionis, K. N. Kannan, and J. H. Panchal, "Quantifying the impact of domain knowledge and problem framing on sequential decisions in engineering design," *Journal of Mechanical Design*, vol. 140, no. 10, 2018.

[84] M. N. Shergadwala, J. H. Panchal, and I. Bilionis, "Quantifying the influence of information sharing about competitor's performance on a participant's sequential design behaviors in design contests," *Journal of Mechanical Design, p. under review*, 2019.

[85] R. P. Smith and S. D. Eppinger, "A predictive model of sequential iteration in engineering design," *Management Science*, vol. 43, no. 8, pp. 1104–1120, 1997.

[86] S. W. Miller, T. W. Simpson, and M. A. Yukish, "Design as a sequential decision process: A method for reducing design set space using models to bound objectives," in *ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2015, pp. V02AT03A020–V02AT03A020.

[87] S. Griffin, N. J. Welton, and K. Claxton, "Exploring the research decision space: the expected value of information for sequential research designs," *Medical Decision Making*, vol. 30, no. 2, pp. 155–162, 2010.

[88] C. Meier, A. A. Yassine, and T. R. Browning, “Design process sequencing with competent genetic algorithms,” *Journal of Mechanical Design*, vol. 129, no. 6, pp. 566–585, 2007.

[89] M. O. Duff and A. Barto, “Optimal learning: Computational procedures for bayes-adaptive markov decision processes,” Ph.D. dissertation, University of Massachusetts at Amherst, 2002.

[90] A. Raina, C. McComb, and J. Cagan, “Learning to design from humans: Imitating human designers through deep learning,” *Journal of Mechanical Design*, vol. 141, no. 11, 2019.

[91] S. Oh, Y. Jung, S. Kim, I. Lee, and N. Kang, “Deep generative design: Integration of topology optimization and generative models,” *Journal of Mechanical Design*, vol. 141, no. 11, 2019.

[92] G. M. Stump, S. W. Miller, M. A. Yukish, T. W. Simpson, and C. Tucker, “Spatial Grammar-Based Recurrent Neural Network for Design Form and Behavior Optimization,” *Journal of Mechanical Design*, vol. 141, no. 12, 2019.

[93] C. Esteban, O. Staeck, S. Baier, Y. Yang, and V. Tresp, “Predicting clinical events by combining static and dynamic information using recurrent neural networks,” in *Healthcare Informatics (ICHI), 2016 IEEE International Conference on.* Ieee, 2016, pp. 93–101.

[94] D. Makris, M. Kaliakatsos-Papakostas, I. Karydis, and K. L. Kermanidis, “Combining LSTM and feed forward neural networks for conditional rhythm composition,” in *International Conference on Engineering Applications of Neural Networks.* Springer, 2017, pp. 570–582.

[95] A. Sharma, “A combined static and dynamic feature extraction technique to recognize handwritten digits,” *Vietnam Journal of Computer Science*, vol. 2, no. 3, pp. 133–142, 2015.

[96] A. Almeida and G. Azkune, “Predicting Human Behaviour with Recurrent Neural Networks,” *Applied Sciences*, vol. 8, no. 2, p. 305, 2018.

[97] A. Lazaric, *Transfer in Reinforcement Learning: A Framework and a Survey.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 143–173. [Online]. Available: https://doi.org/10.1007/978-3-642-27645-3_5

[98] C. Phillips, “Knowledge transfer in markov decision processes,” Citeseer, Tech. Rep., 2006.

[99] N. Mehta, S. Natarajan, P. Tadepalli, and A. Fern, “Transfer in variable-reward hierarchical reinforcement learning,” *Machine Learning*, vol. 73, no. 3, p. 289, Jun 2008. [Online]. Available: https://doi.org/10.1007/s10994-008-5061-y

[100] J. Rajendran, A. Srinivas, M. M. Khapra, P. Prasanna, and B. Ravindran, “Attend, adapt and transfer: Attentive deep architecture for adaptive transfer from multiple sources in the same domain,” 2015. [Online]. Available: https://arxiv.org/abs/1510.02879

[101] F. T. Sunmola and J. L. Wyatt, “Model transfer for markov decision tasks via parameter matching,” in *WORKSHOP OF THE UK PLANNING AND SCHEDULING SPECIAL INTEREST GROUP*, 2006.

[102] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel, “Rl^2: Fast reinforcement learning via slow reinforcement learning,” 2016. [Online]. Available: https://arxiv.org/abs/1611.02779

[103] A. Zhang, H. Satija, and J. Pineau, “Decoupling dynamics and reward for transfer learning,” 2018.

[104] G. Konidaris and A. Barto, “Building portable options: Skill transfer in reinforcement learning,” in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 895–900.

[105] M. H. Goldstein, . Purzer, R. S. Adams, J. Chao, and C. Xie, “The relationship between design reflectivity and conceptions of informed design among high school students,” *European Journal of Engineering Education*, vol. 44, no. 1-2, pp. 123–136, mar 2019. [Online]. Available: https://doi.org/10.1080/03043797.2018.1498458

[106] S. Finger and J. R. Dixon, “A review of research in mechanical engineering design. Part I: Descriptive, prescriptive, and computer-based models of design processes,” *Research in engineering design*, vol. 1, no. 1, pp. 51–67, 1989.

[107] . Purzer, M. H. Goldstein, R. S. Adams, C. Xie, and S. Nourian, “An exploratory study of informed engineering design behaviors associated with scientific explanations,” *International Journal of STEM Education*, vol. 2, no. 1, p. 9, May 2015. [Online]. Available: https://doi.org/10.1186/s40594-015-0019-7

[108] A. Dong, A. W. Hill, and A. M. Agogino, “A document analysis method for characterizing design team performance,” *J. Mech. Des*, vol. 126, no. 3, pp. 378–385, 5/15. [Online]. Available: https://doi.org/10.1115/1.1711818

[109] F. Coley, O. Houseman, and R. Roy, “An introduction to capturing and understanding the cognitive behaviour of design engineers,” *null*, vol. 18, no. 4, pp. 311–325, Aug. 2007. [Online]. Available: https://doi.org/10.1080/09544820600963412

[110] H. A. Simon, “Theories of bounded rationality,” *Decision and organization*, vol. 1, no. 1, pp. 161–176, 1972.

[111] C. Xie, C. Schimpf, J. Chao, S. Nourian, and J. Massicotte, “Learning and teaching engineering design through modeling and simulation on a cad platform,” *Computer Applications in Engineering Education*, vol. 26, no. 4, pp. 824–840, 2018.

[112] R. Gajewski and P. Pieniek, "Building energy modelling and simulations: qualitative and quantitative analysis," *MATEC Web Conf.*, vol. 117, 2017. [Online]. Available: https://doi.org/10.1051/matecconf/201711700051

[113] A. De Mauro, M. Greco, and M. Grimaldi, "What is big data? a consensual definition and a review of key research topics," *AIP Conference Proceedings*, vol. 1644, no. 1, pp. 97–104, May 2022. [Online]. Available: https://aip.scitation.org/doi/abs/10.1063/1.4907823

[114] C. Xie, Z. Zhang, S. Nourian, A. Pallant, and E. Hazzard, "A time series analysis method for assessing engineering design processes using a CAD tool," *International Journal of Engineering Education*, vol. 30, no. 1, pp. 218–230, 2014.

[115] C. Xie, Z. Zhang, S. Nourian, A. Pallant, and S. Bailey, "On the instructional sensitivity of cad logs," *International Journal of Engineering Education*, vol. 30, no. 4, pp. 760–778, 2014.

[116] R. S. Adams, M. Goldstein, S. Purzer, J. Chao, C. Xie, and S. Nourian, "Traversing the barriers to using big data in understating how high school students design," in *Design Computing and Cognition '16*, J. S. Gero, Ed. Cham: Springer International Publishing, 2017, pp. 613–631.

[117] T. M. Kodinariya and P. R. Makwana, "Review on determining number of Cluster in K-Means Clustering," *International Journal*, vol. 1, no. 6, pp. 90–95, 2013.

[118] D. W. Stroock, *An introduction to Markov processes.* Springer Science & Business Media, 2013, vol. 230.

[119] S. Karlin, *A first course in stochastic processes.* Academic press, 2014.

[120] P.-N. Tan, M. Steinbach, and V. Kumar, "Data mining cluster analysis: basic concepts and algorithms," *Introduction to data mining*, 2013.

[121] S. Ayramo and T. Karkkainen, "Introduction to partitioning-based clustering methods with a robust example," *Reports of the Department of Mathematical Information Technology. Series C, Software engineering and computational intelligence*, no. 1/2006, 2006.

[122] V. Chaoji, M. Al Hasan, S. Salem, M. J. Zaki, and Ieee, "Sparcl: Efficient and effective shape-based clustering," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, 2008, pp. 93–102.

[123] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping multidimensional data.* Springer, 2006, pp. 25–71.

[124] H.-P. Kriegel, ger Krö Peer, J. Sander rg, and A. Zimek, "Density-based clustering," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 231–240, 2011.

[125] S. E. Schaeffer, “Graph clustering,” *Computer science review*, vol. 1, no. 1, pp. 27–64, 2007.

[126] V. K. Singh, N. Tiwari, S. Garg, and Ieee, “Document clustering using k-means, heuristic k-means and fuzzy c-means,” in *Computational Intelligence and Communication Networks (CICN), 2011 International Conference on*, 2011, pp. 297–301.

[127] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*. Springer, 2013, vol. 112.

[128] M. Steinbach, G. Karypis, V. Kumar, Others, and Boston, “A comparison of document clustering techniques,” in *KDD workshop on text mining*, vol. 400, 2000, pp. 525–526.

[129] M. Chen, K. Kuzmin, and B. K. Szymanski, “Community detection via maximization of modularity and its variants,” *IEEE Transactions on Computational Social Systems*, vol. 1, no. 1, pp. 46–65, 2014.

[130] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, “On modularity clustering,” *IEEE transactions on knowledge and data engineering*, vol. 20, no. 2, pp. 172–188, 2008.

[131] S. Wagner and D. Wagner, *Comparing clusterings: an overview*. Universität Karlsruhe, Fakultät für Informatik Karlsruhe, 2007.

[132] P. Egan and J. Cagan, “Human and Computational Approaches for Design Problem-Solving BT - Experimental Design Research: Approaches, Perspectives, Applications,” P. Cash, T. Stanković, and M. Štorga, Eds. Cham: Springer International Publishing, 2016, pp. 187–205. [Online]. Available: https://doi.org/10.1007/978-3-319-33781-4-11

[133] D. Boud, R. Keogh, and D. Walker, *Reflection: Turning experience into learning*. Routledge, 2013.

[134] R. Valkenburg and K. Dorst, “The reflective practice of design teams,” *Design Studies*, vol. 19, no. 3, pp. 249–271, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142694X98000118

[135] G. Prudhomme, J. F. Boujut, and D. Brissaud, “Toward reflective practice in engineering design education,” *International Journal of Engineering Education*, vol. 19, no. 2, pp. 328–337, 2003.

[136] Q. Le and T. Mikolov, “Distributed Representations of Sentences and Documents,” in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML’14. JMLR.org, 2014, pp. II–1188–II–1196.

[137] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, “A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks,” in *Proceedings 40th IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2015*, 2015, pp. 5–pages.

[138] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," *Neurocomputing*, vol. 184, pp. 232–242, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231215017671

[139] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[140] "KolmogorovSmirnov Test BT - The Concise Encyclopedia of Statistics." New York, NY: Springer New York, 2008, pp. 283–287. [Online]. Available: https://doi.org/10.1007/978-0-387-32833-1-214

[141] M. Sundermeyer, H. Ney, and R. Schlüter, "From feedforward to recurrent LSTM neural networks for language modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 517–529, 2015.

[142] B. Hanin, "Universal function approximation by deep neural nets with bounded width and relu activations," *arXiv preprint arXiv:1708.02691*, 2017.

[143] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[144] M. Verleysen and D. François, "The curse of dimensionality in data mining and time series prediction," in *International Work-Conference on Artificial Neural Networks*. Springer, 2005, pp. 758–770.

[145] K. Potdar, T. S. Pardawala, and C. D. Pai, "A comparative study of categorical variable encoding techniques for neural network classifiers," *International Journal of Computer Applications*, vol. 175, no. 4, pp. 7–9, 2017.

[146] R. Kohavi, Others, and C. Montreal, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, 1995, pp. 1137–1145.

[147] F. Chollet, *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*. MITP-Verlags GmbH & Co. KG, 2018.

[148] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1, no. 10.

[149] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[150] Z. Sha, Y. Huang, J. S. Fu, M. Wang, Y. Fu, N. Contractor, and W. Chen, "A network-based approach to modeling and predicting product coconsideration relations," *Complexity*, vol. 2018, 2018.

[151] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[152] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[153] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, 2015.

[154] P. J. Werbos *et al.*, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[155] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. [Online]. Available: https://www.aclweb.org/anthology/D14-1179

[156] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[157] M. Suwa and B. Tversky, "What do architects and students perceive in their design sketches? a protocol analysis," *Design studies*, vol. 18, no. 4, pp. 385–403, 1997.

[158] G. Goldschmidt, *Linkography: unfolding the design process.* Mit Press, 2014.

[159] M. H. Rahman, M. Gashler, C. Xie, and Z. Sha, "Automatic clustering of sequential design behaviors," in *ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.* American Society of Mechanical Engineers, 2018, pp. V01BT02A041–V01BT02A041.

[160] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning.* MIT press, 2016.

[161] J. D. Summers and J. J. Shah, "Mechanical engineering design complexity metrics: size, coupling, and solvability," *Journal of Mechanical Design*, vol. 132, no. 2, p. 021004, 2010.

[162] J. S. Gero, "Design prototypes: a knowledge representation schema for design," *AI magazine*, vol. 11, no. 4, p. 26, 1990.

[163] J. S. Gero and U. Kannengiesser, "The situated function–behaviour–structure framework," *Design studies*, vol. 25, no. 4, pp. 373–391, 2004.

[164] T. M. Kodinariya and P. R. Makwana, "Review on determining number of cluster in k-means clustering," *International Journal*, vol. 1, no. 6, pp. 90–95, 2013.

[165] S. Yadav and S. Shukla, "Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification," in *Advanced Computing (IACC), 2016 IEEE 6th International Conference on.* IEEE, 2016, pp. 78–83.

[166] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[167] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[168] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[169] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[170] S. Caton and C. Haas, "Fairness in machine learning: A survey," 2020. [Online]. Available: https://arxiv.org/abs/2010.04053

[171] H. Wu, A. Ghadami, A. E. Bayrak, J. M. Smereka, and B. I. Epureanu, "Impact of Heterogeneity and Risk Aversion on Task Allocation in Multi-Agent Teams," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7065–7072, 2021.

[172] D. H. Gensch and W. W. Recker, "The Multinomial, Multiattribute Logit Choice Model," *Journal of Marketing Research*, vol. 16, no. 1, pp. 124–132, feb 1979. [Online]. Available: https://doi.org/10.1177/002224377901600117

[173] C. Xie, C. Schimpf, J. Chao, S. Nourian, and J. Massicotte, "Learning and teaching engineering design through modeling and simulation on a CAD platform," *Computer Applications in Engineering Education*, vol. 26, no. 4, pp. 824–840, 2018.

[174] J. S. Gero and U. Kannengiesser, "The Function-Behaviour-Structure Ontology of Design BT - An Anthology of Theories and Models of Design: Philosophy, Approaches and Empirical Explorations," A. Chakrabarti and L. T. M. Blessing, Eds. London: Springer London, 2014, pp. 263–283. [Online]. Available: https://doi.org/10.1007/978-1-4471-6338-1_13

[175] U. Kannengiesser, J. S. Gero, C. M. De Smet, and J. A. Peeters, "An ontology of computer-aided design," in *Computer-Aided Design Research and Development*. Nova Science Publishers, 2009.

[176] M. H. Rahman, S. Yuan, C. Xie, and Z. Sha, "Predicting human design decisions with deep recurrent neural network combining static and dynamic data," *Design Science*, vol. 6, p. e15, 2020.

[Online]. Available: https://www.cambridge.org/core/article/predicting-human-design-decisions-with-deep-recurrent-neural-network-combining-static-and-dynamic-data/097456E3CE09F11435F535B507AE9B8B

[177] A. Burnap, Y. Ren, R. Gerth, G. Papazoglou, R. Gonzalez, and P. Y. Papalambros, “When Crowdsourcing Fails: A Study of Expertise on Crowdsourced Design Evaluation,” *Journal of Mechanical Design*, vol. 137, no. 3, mar 2015. [Online]. Available: https://doi.org/10.1115/1.4029065

[178] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, and W. Zhang, “A comparative study on transformer vs RNN in speech applications,” in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, dec 2019. [Online]. Available:

A Description of Design Experiment

A.1 Energyplus home

A.1.1 Experiment instruction

Overview

This is an experiment of studying design decision-making behaviors in engineering systems design. The University of Arkansas has provided funds for this research. You are eligible to participate in this experiment if you are 18 years of age or above. Please note that participating in this experiment is voluntary. If you decide to participate, no identifying information will be collected and your data will be kept strictly confidential. The experiment will last 2 hours. Should you need assistance of any kind, please raise your hand and an experimenter will come to you. Disqualification can occur if any of the following rules are broken. In the event of disqualification, you will be asked to leave and you will not be paid. We expect and appreciate your cooperation.

- Please remain silent and do not look at other peoples work.
- Please do not talk, laugh, exclaim out loud, etc.
- You must not access Internet or use your phone during the experiment.

NOTE: In emergency, stop the experiment right away and follow the university instructions.

Pre-Session

This session will last around 30 minutes for you to **practice** the Energy3D software. To start the experiment, click the **Experiment.bat** file on the desktop and follow the instruction. You need to fill a questionnaire first. Once you finish it, close the browser and you will be asked to press any key to proceed to the practice session. During the practice, a tutorial sheet is available to you to refer to the key operations and terminologies in Energy3D. You are recommended to explore Energy3D as much as you can in this session. By the end of pre-session, you will be asked to close the current Energy3D window and get prepared for the next session.

In-Session

After closing the practice window, press any key to start the main design session. During this session, you will take a design challenge to build your solarized energy-plus home. You will be given an instruction sheet that outlines the design objective and the design requirements. The tutorial sheet will be still available to you in this session. *NOTE: Do NOT change file name in the middle of the design. Do NOT save as your project. In case of errors pop up and you cannot proceed, remember* *to save the file before closing your project. Then raise your hand, you will be directed to open the same file to continue your design.*

After-Session

At the end of the session, run the annual net energy (ANE) and cost analysis of your final design before you close Energy3D. Put the tutorial sheet and instruction sheets beside your laptop. Sign out at the front of the room. You will be paid in private and in cash.

Rewarding Rule

Your final payoff consists of two parts: 1) the time rewards and 2) the design rewards. For the time rewards, you will be paid $15, if you exploit the entire 2 hours. You are allowed to sign out and leave early, but payment will be prorated based on the time you have spent. The design rewards are proportional to your final design, specifically, the *ANE generated per dollar cost.*

A.1.2 Design a solarized energy-Plus home

Design statement: You are going to design a solarized energy-plus home (see an example in Figure 1) using Energy3D for your client in Dallas, Texas. **Your budget is $200,000**

Table A.1: Statistical t-test on the difference between the prediction accuracy of the baseline models and the models developed in the two case studies

	Items	**Specifications**
Build height	Story	1
	Height of wall	$\geq 2.5m$
Roof	Roof style	Pitched
Window	Number of windows	≥ 4
	Size of window	$\geq 1.44m^2$
Door	Number of doors	1
	Size of door (Width Height)	1.2×1.2
Solar panel	Solar panel placement	On roof only
	Distance between the ridge to the solar panel (i.e., and in Figure 1)	$x \geq 0$ and $y \geq 0$
	Solar Panel Orientation	Portrait

Design objective: To **maximize the annual net energy (ANE)** of your house given the budget.

Design requirements: The full list of the requirements that you must meet in your design are summarized in Table 1.

Note: You can only use solar panel and/or solar panel rack (preferred) in this project. Other solar energy harvesting devices, e.g., the planar mirror, parabolic dish, etc. are not allowed to use in this project.

A.1.3 Tutorials for energy-plus home design challenge

Quick Tips:

1. To delete any existing feature, just select that feature and press Delete key.
2. To undo an operation, press Crtl +Z.
3. If you want to get additional information of an artifact (e.g., wall, roof, etc.), you need to select the object and right click the mouse. But some parameters of an artifact do not affect the design cost and annual net energy (ANE) e.g. tint of window, color of door etc. This tutorial enlists parameters that have direct relation with design cost or ANE.
4. You can resize an object by dragging the white control point. You can move an object by dragging the orange control point.
5. If you quickly access the information of a design artifact on the right-hand-side panel.
6. If you encounter any errors, please ignore them and your design will not be affected. In case your design cannot be proceeded, please save the file and reopen it to continue your work.

   Energy3D is a simulation-based computer-aided design (CAD) tool for green buildings and power stations that harness renewable energy to achieve sustainable development. The tutorial provides you with quick reference to basic and key operations in Energy3D for your design and analysis of an energy-plus home. Please feel free to explore other operations while you are practicing.

Foundation Operation

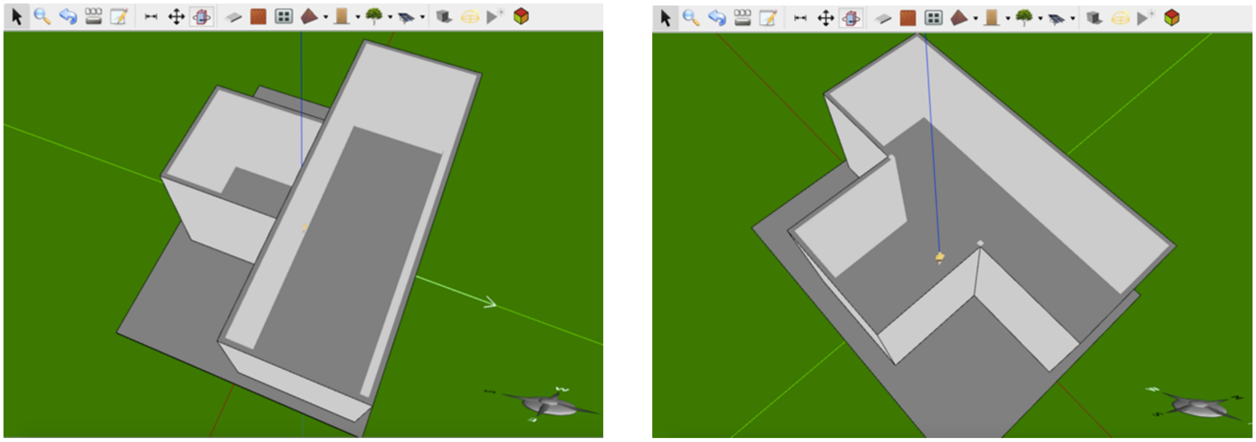
Add foundation: By default, you will see a foundation the time when you create a new project. You can add more foundations, but you have to make sure that each foundation carries a building.

Figure A.1: Add foundation

Edit foundation: If you click on the foundation, you can see its size in the foundation tab (right of the main interface; just below the project tab). To resize the foundation, you can drag the control point of the foundation to a size you need. To edit precisely, you can right click on the foundation. From the pop-up menu, you can enter your desired size values. You can move the foundation by dragging the orange control point.

Wall Operation

Add wall:You can add walls by clicking the draw wall button from the taskbar. The grid lines on the platform visually guide you to set the position, direction, and length of a wall. You can also add a wall in the 2D mode just like the floor plan. It allows



(a) Wrong procedure

(b) Correct procedure

Figure A.2: Add wall

you to add the wall from the top view. To get into the 2D mode, go to the view option of the taskbar and select 2D top view. You can use shortcut Ctrl +T. While adding the walls, make sure that all the walls are connected to each other to form a closed space. See the correct operation of adding walls in Figure 3.

Edit wall:By right clicking on the wall, you can go to the other options and change the wall setting with your requirement (see Figure 4).

Change height:By right clicking on the wall and selecting the height option, you can change the height of the walls, either one at a time or all the walls at the same time.

Roof Operation

Add roof: Energy3D allows you to choose different roof styles for your home. For a

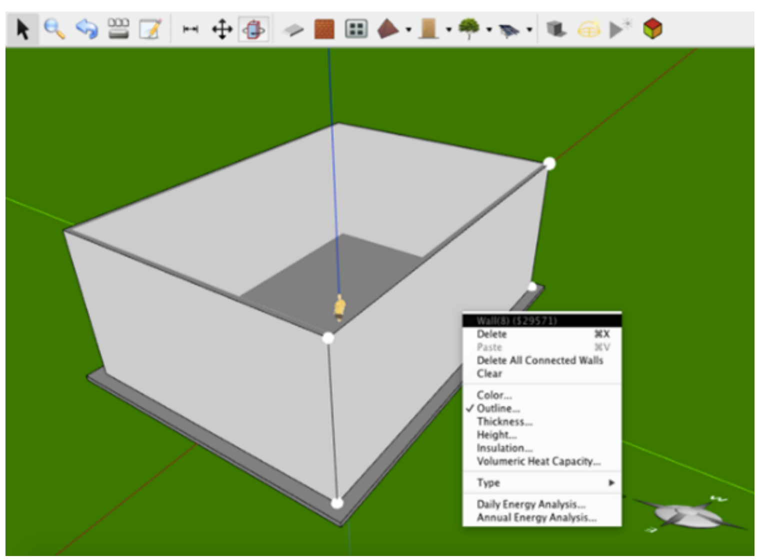


Figure A.3: Edit wall

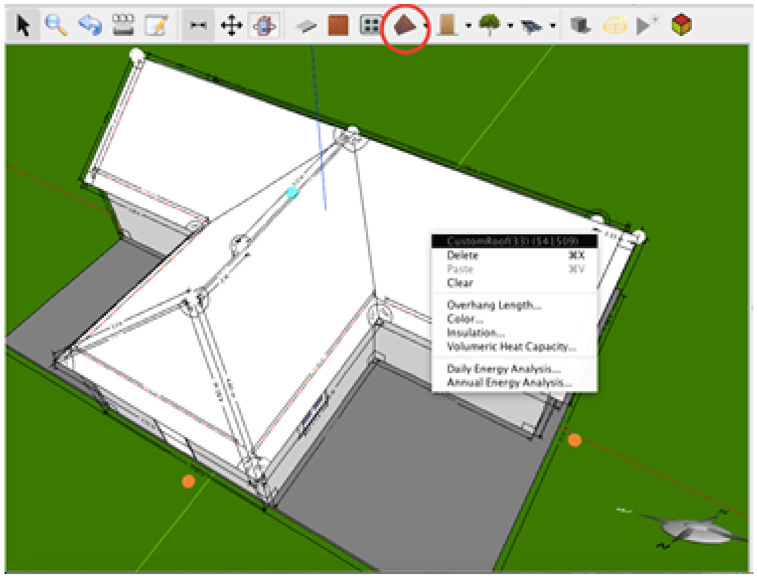


Figure A.4: Change the insulation of the roof

simple design, you can use the hip roof. If you have a complicated floor plan, you can choose custom roof.

Height of the roof:By dragging the apex control point (the blue point in Figure 5), you can change the height of the roof.

Roof angle:Roof angle is related to roof design. You can change the roof angle by dragging the white control point.

Insulation of the roof:You can change the insulation type of the roof by right-clicking on the roof and going to the Insulation option.

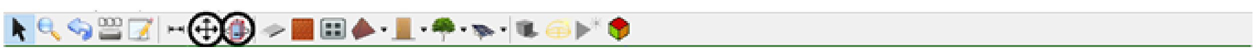


Figure A.5: Resize, move and rotate

Resize, Move and Rotate

This feature allows you to focus on designing the shape of the building without worrying about its exact size at the beginning. After you complete the shape design, you can then resize the building in any ways. You can also move the whole building with the same button. For resizing, use the white control point and for moving, use the orange control point. If you want to rotate the building along with the foundation, keep clicking the Rotate button until you get the desired position.

Window Operation

Add window:To add a window, click the window shape button on the taskbar and follow the grid line to place the window. You can copy and paste a window.

Solar heat gain coefficient, insulator:To change the solar heat gain coefficient (SGHC) of a window, just right click on the window and select Solar Heat Gain Coefficient. For insulator parameter, the procedure is same.

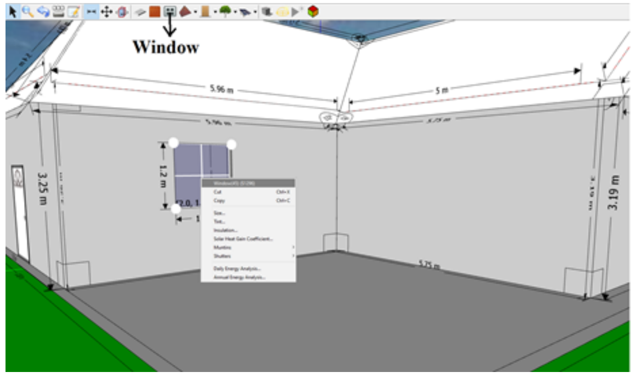


Figure A.6: Change the parameters of the window

Solar heat gain coefficient, insulator:To change the solar heat gain coefficient (SGHC) of a window, just right click on the window and select Solar Heat Gain Coefficient. For insulator parameter, the procedure is same.

Door Operation

Add door:To add a door, click the door shape button on the taskbar and follow the grid line to place the window. You can copy and paste a door.

Door insulation:You can change the insulation type of the door by right clicking on the door and selecting Insulation option.

Heliodon Simulator, Show Shadow and Irradiance Heat Map

Show heliodon:You can observe the sun path using the heliodon simulator (see Figure 8).

Show shadow and animate sun:If you click the shadow button and the sun animation button on the Task Bar, you can watch how an object casts shadow on the ground and how sunlight shines into the house as the sun moves across the sky.

Show irradiance heat map:You can use the solar radiation simulator to evaluate

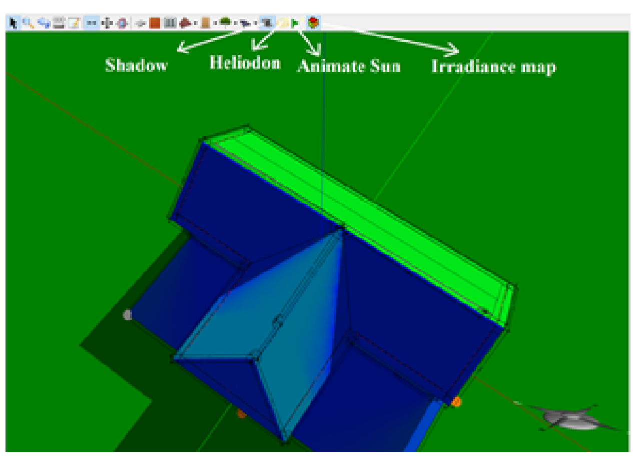


Figure A.7: Heliodon, shadow and heat map

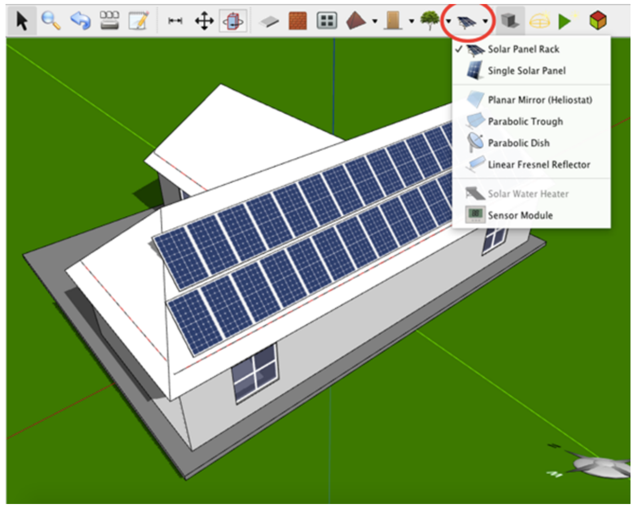


Figure A.8: Add solar panel

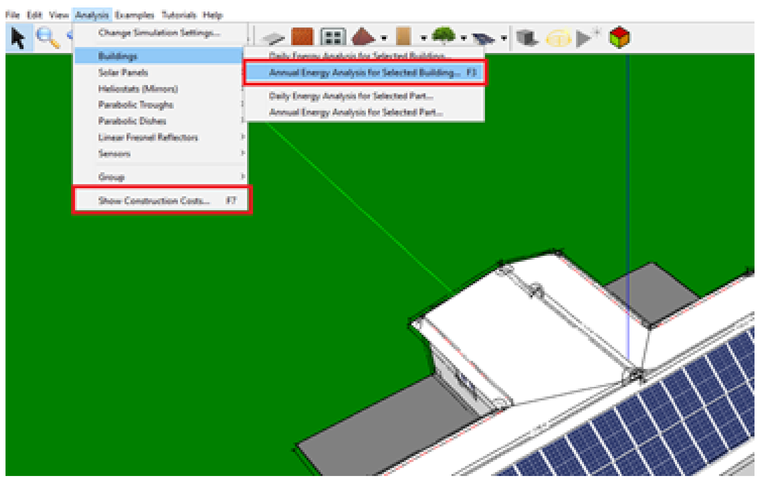


Figure A.9: Annual Net Energy and Design cost

the daily solar potential or daily-absorbed solar energy of your house. Blue color in a heat map represents a low value of solar heat and red represents a high value of solar heat.

Solar Panel Operation

Add solar panel:You can add a single solar panel or a solar panel rack by clicking the solar panel button on the taskbar and select from the pull-down menu. If you select solar rack, you have to drag white control point to expand it.

Solar panel size:You can adjust the size by clicking on the solar panel. From the menu, select solar panel properties and go to the size option.

Solar cell efficiency:You can also change the solar cell efficiency by following the same step as you size the solar panel, but go to solar cell efficiency instead. Increasing cell efficiency increases solar gain, but it will cost more.

Annual Net Energy

To analyze the annual net energy (ANE) of your house, first select the foundation of your house, go the analysis tab and under building menu select Annual Energy Analysis for Selected Building option. You can also press F3 to start the ANE analysis. Do not use other options for ANE analysis.

Show Design Cost

You can get the total design cost of your building by clicking the show construction cost in the analysis tab. You can also press F7 to show the cost. Energy3D also provides an itemized cost of the house for every component.

Glossary:

A.2 Parking lot

A.2.1 Experiment instruction

Overview

This is an experiment of studying design decision-making behaviors in engineering systems design. The University of Arkansas has provided funds for this research. You are eligible to participate in this experiment if you are 18 years of age or above. Please note that participating in this experiment is voluntary. If you decide to participate, no identifying information will be collected and your data will be kept strictly confidential. The experiment will last 2 hours. Should you need assistance of any kind, please raise your hand and an experimenter will come to you. Disqualification can occur if any of the following rules are broken. In the event of disqualification, you will be asked to leave and you will not be paid. We expect and appreciate your cooperation.

NOTE: In emergency, stop the experiment right away and follow the university instructions.

Pre-Session: This session will last around 30 minutes for you to practice the Energy3D software. To start the experiment, click the Experiment.bat file on the desktop and follow the instruction. You need to fill a questionnaire first. Once you finish it, close the browser and you will be asked to press any key to proceed to the practice session. During the practice, a tutorial sheet is available to you to refer to the key operations and terminologies in Energy3D. You are recommended to explore Energy3D as much as you can in this session. By the end of pre-session, you will be asked to close the current Energy3D window and get prepared for the next session.

In-Session

After closing the practice window, press any key to start the main design session. During this session, you will take a design challenge to build your **Solarized UARK Parking Lot**. You will be given an instruction sheet that outlines the design objective and the design requirements. The tutorial sheet will be still available to you in this session.

- Do NOT change file name. Do NOT save as your project. In case of errors pop up, remember to save the file before closing your project.
- Whenever you change any parameters of the solar panel, you MUST do the annual yield analysis (AYE) (F4) and cost analysis (F7). You are also recommended to the AYE and cost analysis as frequent as possible during the whole design process.
- There are other design variables related to the solar panel and the solar rack, but only the ones mentioned in the Tutorial Sheet are allowed to change.

After-Session

: At the end of the session, run the AYE (F4) and cost analysis (F7) of your final design before you close Energy3D. Put the tutorial sheet and instruction sheets beside your laptop. Sign out at the front of the room. You will be paid in private and in cash.

Rewarding Rule

Your final payoff consists of two parts: 1) the time rewards and 2) the design rewards. For the time rewards, you will be paid $15, if you exploit the entire 2 hours. You are allowed to sign out and leave early, but payment will be prorated based on the time you have spent. The design rewards are proportional to your final design, specifically, the **ANE generated per dollar cost**.

A.2.2 Solarize UARK parking lots

Design statement: You are going to build a solarized parking lot for the University of Arkansas using Energy3D. See an example in Figure 1. Your budget is **$1.5 million** including the construction and 20-year maintenance.

Design objective: To maximize the annual yield energy of your solarized parking lot given the budget. Please refer to the tutorial sheet for doing annual yield energy and cost analysis.

Design requirements: The full list of the requirements that you must meet in your design are summarized in Table A.2.

- Please remain silent and do not look at other peoples work.
- Please do not talk, laugh, exclaim out loud, etc.
- You must not access Internet or use your phone during the experiment.

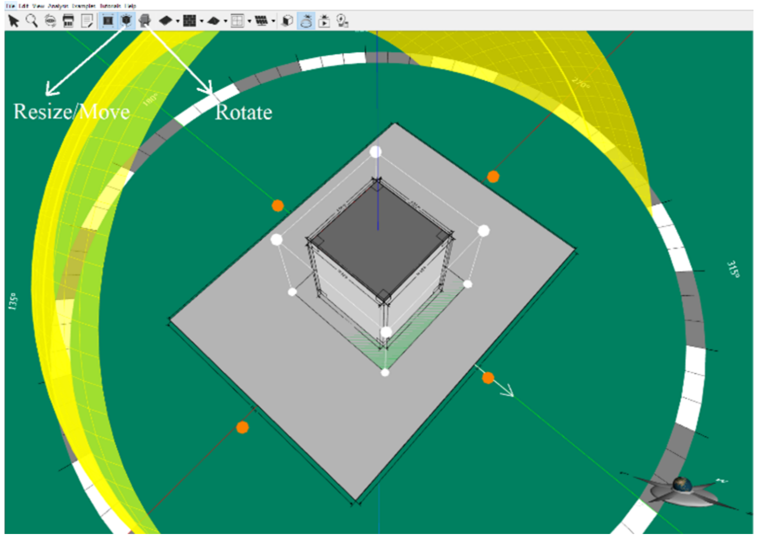


Figure A.10: Resize, move and rotate

Table A.2: Design requirements for parking lot

	The requirements	Constraints
Solar Panel	Tilt angle	20 degree *Notes:* Increasing the tilt angle may impede vehicles from entering the parking spot (see Figure 1). The height of one side decreases 0.08m for every degree of tilting angle increased.
Solar Rack	Solar panel rack	Should not produce any hindrance to the pedestrian zone and driveways
	Base height	3.5 m
	The pole of rack	Should be placed along with the parking lot line marker. Try to place the pole as much closer as to the parking lot mark.

A.2.3 Tutorial on Energy3D

Quick Tips:

This tutorial is used for the project of Solarize the UARK Parking Lots using Energy3D software. Energy3D is a simulation-based computer-aided design (CAD) tool for green buildings and power stations that harness renewable energy to achieve sustainable development. The tutorial provides you with quick reference to basic operations in Energy3D to solarize and analysis the annual yield of your parking lot.

Solar Panel Operation

Add solar panel rack: You can add solar panel rack by clicking the solar panel button (Figure 1).

Solar rack size: If you select solar rack, you can drag the white control points to size the panel you want. You can also right click on the solar panel, and select the Size option to enter exact values.

Move and Rotate: For moving the solar panel, you need to select it first. Then click the orange control point and drag it to you desired position. If you want to rotate the solar panel along with the foundation, keep clicking the Rotate button until you get the desired position (see Figure 1). You can also rotate from Azimuth option which is described in Azimuth section.

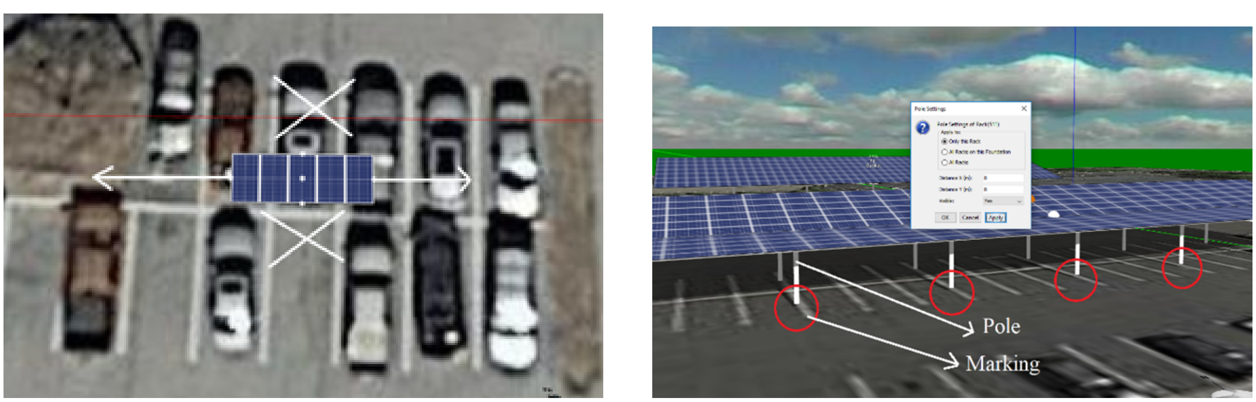
Solar panel tilt angle: The solar panel, if is properly tilted, can increase the amount of solar radiation. The optimum tilt angle depends on the location and height of site. To change the tilt angle, right click on the solar panel and select either the Fixed Tilt Angle option or the Seasonally Adjusted Tilt Angels option. You can use either one. When sizing the solar panel, you have to extend the length in the parallel with the parking marker as shown in Figure 2 a) to tilt the solar panel properly.

Pole settings of the solar racks: You can set the number of the pole of the rack by right clicking the solar panel and selecting Pole Settings option. You should select the pole setting in such a way that it aligns with the line marker of the corresponding parking lot. (See Figure 2 b)

Azimuth: To place the solar panel in a proper direction, you may need to rotate it. Azimuth option can be used to rotate the solar panel. To rotate, right click on the solar panel and set the value of azimuth. You can also rotate solar using the rotate button from the toolbar.

Solar panel base height: You can change the height of the solar panel by right clicking on the solar panel and setting the base height. The height should be minimum so that pedestrian and the vehicle can easily move.

Solar panel layout: There are different layouts for solar panel design (e.g., tilted



(a) Wrong procedure (b) Correct procedure

Figure A.11: Solar panel settings

solar panel, butterfly solar panel, etc.). To create an arrangement, first, select a rack and place it in accordance with the parking lot mark. You can choose two racks and put them sidewise. By adjusting the tilting angle of the racks, you can achieve an optimal layout for a specific parking lot. Make sure that there is a minimum gap between the edges of the solar racks (see Figure 3)

Solar cell efficiency: You can change the solar cell efficiency by right clicking on the solar, go to Change Solar Panel Properties, and then go to Solar Cell Efficiency to set values.

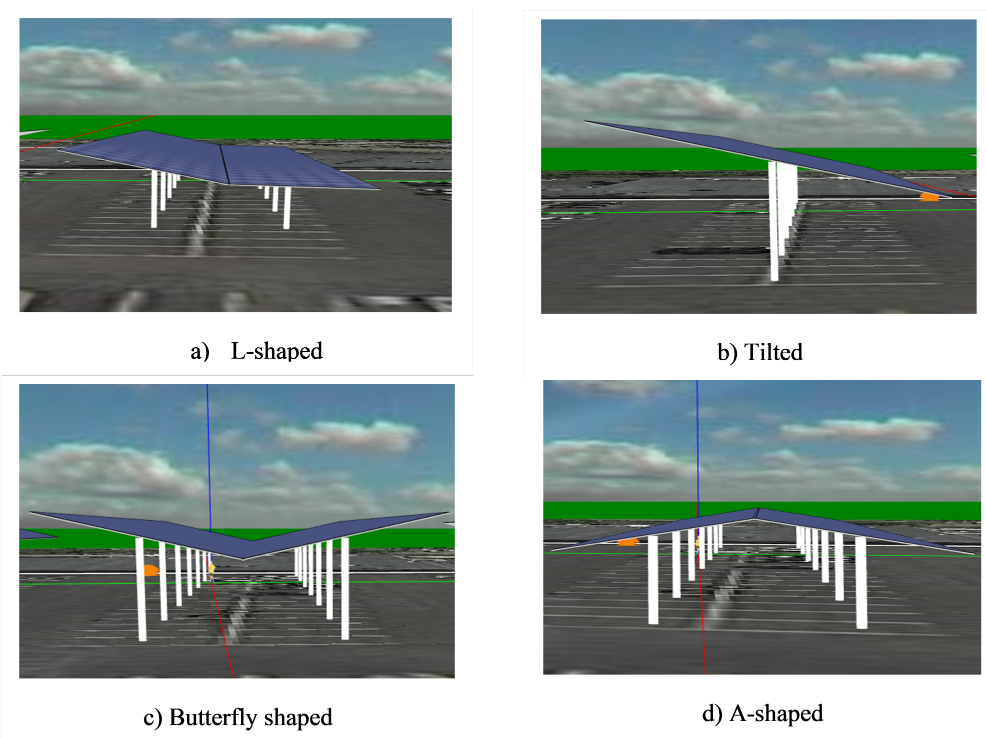
Nominal operating cell temperature: Increasing cell temperature reduces solar cell efficiency. So, it is important to know the expected operating cell temperature. You can change the nominal operating cell temperature between 330C to 580C. To make a change, right click on the solar panel and go to Change Solar Panel Properties to find nominal operating cell temperature.

Heliodon Simulator, Show Shadow and Irradiance Heat Map

Show heliodon You can check the sun path using Heliodon simulator (Figure 4).

Show shadow and animate sun If you click the shadow button and the sun animation button on the task bar, you can watch how an object casts shadow on the ground and how sunlight shines into the ground and the sun moves across the sky.

Show irradiance heat map You can use the solar radiation simulator to evaluate the daily solar potential or daily-absorbed solar energy of the structure. Blue color in



a) L-shaped

b) Tilted

c) Butterfly shaped

d) A-shaped

Figure A.12: Different layouts of solar panel rack a heat map represents a low value of solar heat and red represents a high value of solar heat.

Annual Yield Analysis

To analyze the annual yield energy, first select the foundation on which you have built your structure, then go the analysis tab, under the Solar Panels menu you can select Annual Yield Analysis of Solar Panels option (see Figure 5). You can also press F4 to start the annual yield analysis. Do not use other options. You will be asked Do you want to keep the results of this run? Please choose Yes so that you can compare this result with you next run.

Design Cost You can get the total design cost of your parking lot by looking at the side bar of the main design window (Figure 6). If the cost exceeds the given budget, the total cost turns into yellow. So, you can keep track your cost during the design. You can also use F7 to access the cost information.

A.2.4 Solarize UARK parking lots

Participants were given several instructional documents to guide them throughout the design challenge. Below are the main goals and constraints that were outlined to participants. The instructions seen below were created by the research team specifically for the design challenge that was described in the present study.

Design statement: In this project, you will utilize a predefined rooftop space and parking lot area to arrange the solar panels (Figure A.14). The projects lifetime is

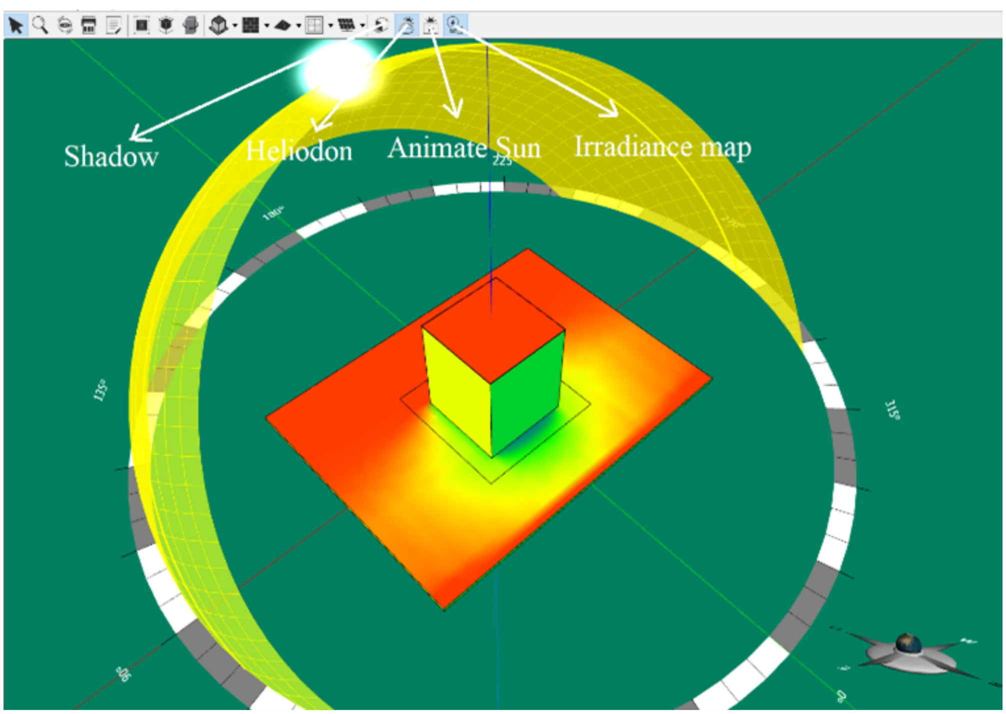


Figure A.13: Heliodon, shadow and heat map set to be 25 years. The university has contracted with a utility company to sell the generated electricity at a guaranteed price of 18 cents per kWh for the next 25 years. The cleaning and maintenance cost per solar panel is $2 per year for each solar panel. The rack for mounting the solar panels costs $20 per solar panel. The interest rate on the bank loan to purchase and install the solar panels, racks, and supporting systems is fixed at 2.95% for 25 years. These parameters have been set in the Aladdin model.

Design objective: Generate more than 1,000,000 kWh of electricity per year with a payback period shorter than 10 years.

Types of Solar Panels The university has negotiated with three manufacturers to offer three different types of solar panels, as shown in the following table (Table A.3). You must select **one and only one** of them for your project. The prices listed in the table have factored in the insurance cost, the installation cost, and other costs.

Rooftop Constraints

1. **Budget Limit:** The total upfront cost of the solar panels, racks, and supporting systems must not exceed $700,000.
2. **Installation Location:** The solar panels must be installed on and within the roofs. Chimneys, vents, and heating and cooling equipment on the roofs must be left open and accessible.

Parking Lot Constraints

1. **Budget Limit:** The total upfront cost of the solar panels, racks, and supporting systems must not exceed $1.2M.
2. **Installation Location:** The solar panels must be installed on and within the predefined parking areas. Trees, obstacles, and boundaries must be avoided. Do

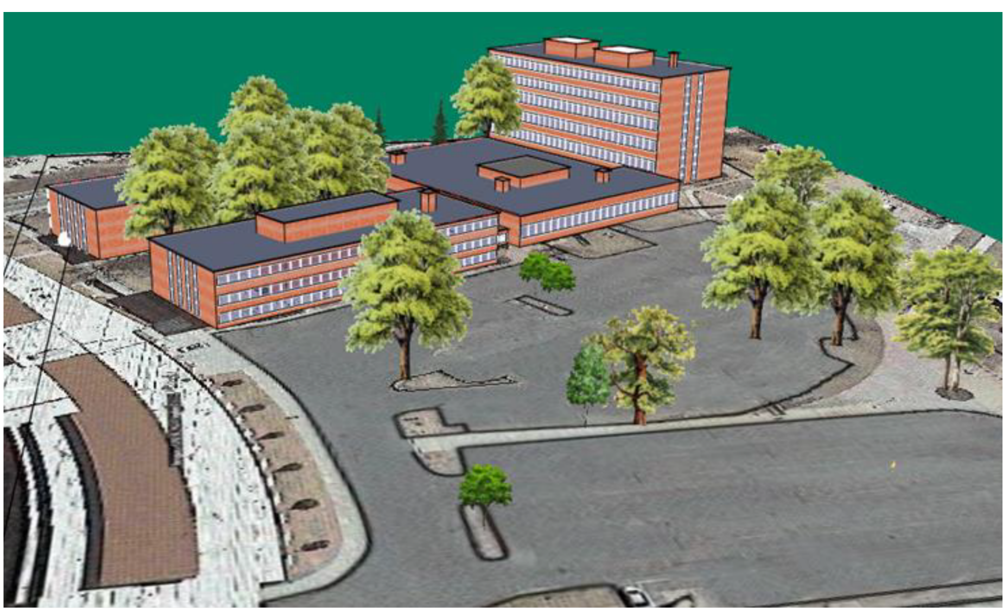


Figure A.14: The CAD environment that participants completed the design challenge in

NOT use any reference parking lines (i.e., from current parking lot on Google Maps) to place the solar panels. The parking lines will be drawn according your design of the layout.

3. **Pole Height and Tilt Angle:** There must be enough space for a car to park under the solar panel. The base height should be $\geq$ 3.5m when the tilt angle of solar panel is 20 degrees. If the tilt angle is increased or decreased, the base height must be adjusted accordingly. **Be aware that pole height influences cost.**
4. **Stall Width and Aisle Width:** There must be enough space to enter and exit a car into the lot. Stall width should be around 5.25m - 6m and aisle width should be at least 7.8m. See Figure A.15 for a visual example.

Parking Lot Constraints During this engineering design activity, you should try changing the following variables to optimize the return on investment so that you can achieve the shortest payback period.

Rooftop and Parking Lot

1. **Type of Solar Panel:** Different solar panels have different efficiencies, dimensions, and costs. A panel with a higher efficiency will generate more electricity. A larger panel will take up more space. A more expensive panel will result in a longer payback period, but it may retain a higher value beyond the project lifetime.
2. **Solar Panel Orientation:** An optimal orientation will allow the solar panel to face the sun and receive more energy during the day. The orientation includes the azimuth and tilt angle of the solar panel.

Rooftop Specific

1. **Solar Panel Location:** The solar panels should be away from the tall structures on the roof, as they may cast shadow on the panels that are too close to them. This will reduce panel output at certain times throughout the day.

Table A.3: Solar panel models to choose from.

Manufacturer	Model	Cost per Panel ($)	Power Output (Watt)	Solar Cell Efficiency	Dimensions
SunPower	SPR-X22-370	$590.00	370W	22.7%	1.04m×1.55m
Yingli	YL375D-36b	$589.00	375W	19.29%	0.99×1.96m
Canadian Solar	CS6X-355P-FG	$529.00	355W	18.18%	0.99m×1.96m

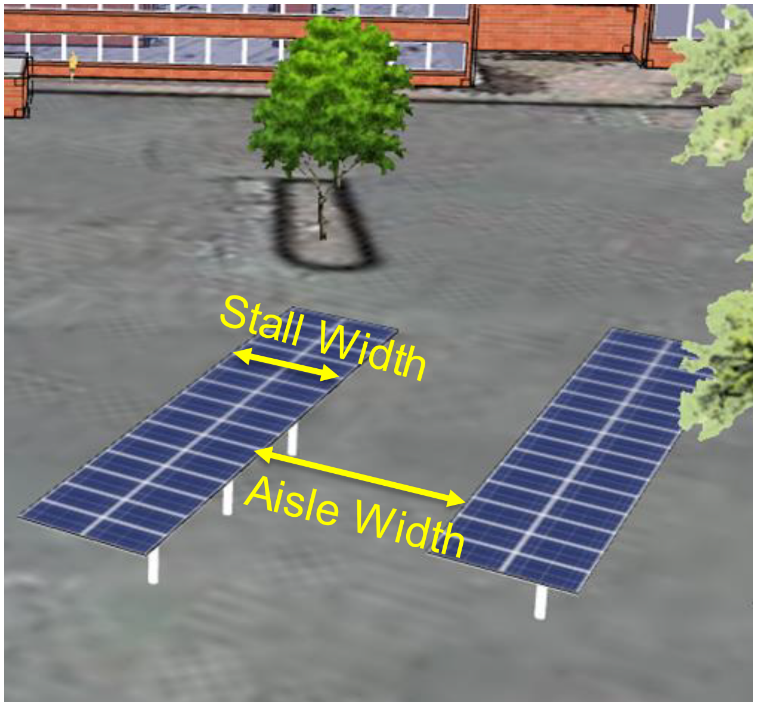


Figure A.15: Stall and aisle width

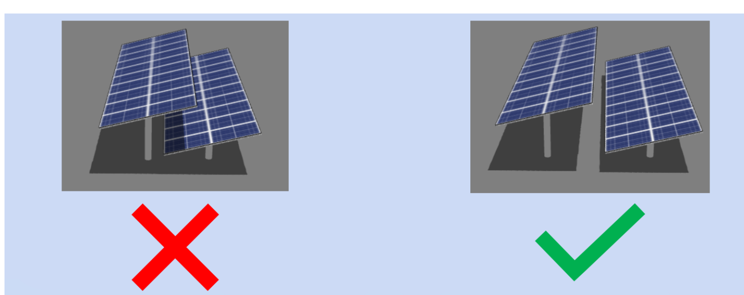


Figure A.16: Participants were advised to not have their solar panels overlap

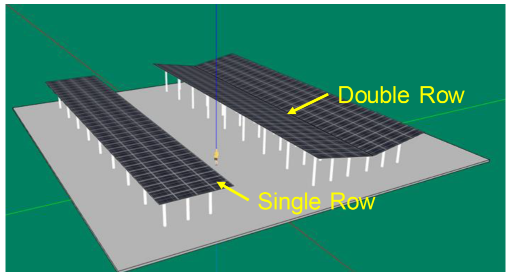


Figure A.17: A single row of panels compared to a double row

2. **Distance Between Adjacent Rows:** The distance between adjacent rows of tilted solar panels will affect their output, as one row may cast shadows on the adjacent panels. See Figure A.16.

Parking Lot Specific

1. **Solar Panel Location:** The solar panels should be away from the tall structures as they may cast shadow on the panels that are too close to them, reducing their output at certain time of the day. The location includes the length and orientation of the solar panel.
2. **Pole Height:** The pole height can be varied with different length. However, the height must be high enough so that cars may park under them. The lowest edge of your solar panel must be at least 3.5m high.
3. **Tilt Angle vs Pole Height:** The tilt angle of the solar panel should be set in such a way so that it can produce maximum solar energy.
4. **Double Row vs Single Row:** Based on the parking space, you can choose either the double row or single row. See Figure A.17 for an example of both methods.

Documentation Review the outputs carefully to check whether your design meets each of the criteria and constraints. You wont be able to get everything right at first try. Try changing the type of solar panels, the layout of the array, and their orientation. Compare the outputs after each change to see whether the results improve. Make informed decisions for your next change based on the analysis results, and try not to guess.

At the end of each iteration, complete the Design Answer Sheet and Design Report. Select one final design to present in your report and justify your recommendation with the results of your analyses.

A.2.5 Record Sheet

Date: ____________________ **Seat No.** ____________________

Iteration	1	2	3	4	5	6	7	8	9	10
Annual Net Energy (ANE)										
Design Cost										
Iteration	11	12	13	14	15	16	17	18	19	20
Annual Net Energy (ANE)										
Design Cost										
Iteration	21	22	23	24	25	26	27	28	29	30
Annual Net Energy (ANE)										
Design Cost										
Iteration	31	32	33	34	35	36	37	38	39	40
Annual Net Energy (ANE)										
Design Cost										
Iteration	41	42	43	44	45	46	47	48	49	50
Annual Net Energy (ANE)										
Design Cost										

You Final Design

The ANE: ____________________________

Design Cost: ___________________________

Systems Integration and Design Informatics Laboratory

Figure A.18: Record sheet

B All Publication Published, Submitted, and Planned

B.1 Journals

1. Rahman, M. H., Schimpf, C., Xie, C., and Sha, Z. (2019). "A Computer-Aided Design Based Research Platform for Design Thinking Studies." ASME.J. Mech. Des. December 2019; 141(12): 121102.https://doi.org/10.1115/1.4044395
2. Rahman, M. H., Yuan, S., Xie, C. and Sha, Z. (2020) Predicting human design decisions with deep recurrent neural network combining static and dynamic data, Design Science. Cambridge University Press, 6, p. e15. doi: 10.1017/dsj.2020.12.
3. Rahman, M. H., Xie, C., and Sha, Z. (2021). "Predicting Sequential Design Decisions Using the Function-Behavior-Structure Design Process Model and Recurrent Neural Networks." ASME.J. Mech. Des. August 2021; 143(8):081706.https://doi.org/10.1115/1.4049971
4. Clay, J., Li, X., Rahman, M. H., Zabelina, D., Goldstein, M.H, Xie, C. and Sha, Z. From Design Cognition to Design Performance: A Clustering-Based Correlation Analysis, Design science. Cambridge University Press (Under review).
5. Rahman, M. H., Bayrak, A. E. and Sha, Z. Empirical Evidence and Computational Assessment on Design Knowledge Transferability, Design Science. Cambridge University Press (Under review)

B.2 Conference

1. Rahman, MH, Gashler, M, Xie, C, & Sha, Z. "Automatic Clustering of Sequential Design Behaviors."Proceedings of the ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.Volume 1B: 38th Computers and Information in Engineering Conference. Quebec City, Quebec, Canada. August 2629, 2018. V01BT02A041. ASME.https://doi.org/10.1115/DETC2018-86300
2. Rahman, MH, Xie, C, & Sha, Z. "A Deep Learning Based Approach to Predict Sequential Design Decisions."Proceedings of the ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference.Volume 1: 39th Computers and Information in Engineering Conference. Anaheim, California, USA. August 1821, 2019. V001T02A029. ASME.https://doi.org/10.1115/DETC2019-97625
3. Rahman, MH, Xie, C, & Sha, Z., Design Embedding: Representation Learning of Design Thinking for Clustering Design Behaviors,ASME 2021 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Virtual Conference, Aug. 17-20, 2021.

4. Rahman, M. H., Bayrak, A. E. & Sha, Z. (2022), A reinforcement learning approach to predicting human design actions using a data-driven reward formulation, in Proceedings of the Design Society: DESIGN Conference, Vol. 1, Cambridge University Press, pp. 17091718.
5. Clay, J.Z., Rahman, M.H., Zabelina, D.L., Xie, C., Sha, Z. (2022). The Psychological Links Between Systems Thinking and Sequential Decision Making in Engineering Design. In: Gero, J.S. (eds) Design Computing and Cognition20. Springer, Cham. https://doi.org/10.1007/978-3-030-90625-2_4
6. Clay, J., Li, X., Rahman, M. H., Zabelina, D., Xie, C. and Sha, Z. (2021). Modeling and Profiling Student Designers' Cognitive Competencies in Computer-Aided Design, Proceedings of the Design Society. Cambridge University Press, 1, pp. 21572166. doi: 10.1017/pds.2021.477

B.3 Posters presentation

1. Rahman, M.H., Sha, Z., Modeling Sequential Decision Making in Engineering Systems Design, CIE Graduate Research Poster Competition, ASME 2018 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Quebec City, Canada. 2018.
2. Rahman, M.H., Sha, Z., A Deep Learning-Based Framework to Predict Sequential Design Decisions, CIE Graduate Research Poster Competition, ASME 2019 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Anaheim, CA. 2019.
3. Rahman, M.H., Xie, C., Sha, C., Towards Building an AI-Integrated Computer-Aided Design Platform for Design Research, NSF Research Poster Competition, The International Mechanical Engineering Congress and Exposition (IMECE), Salt Lake City, UT, Nov. 8-14, 2019. Poster.