Concept of Interactive Machine Learning in Urban Design Problems

Artem M. Chirkin
ETH Zürich
Wolfgang-Pauli-Str. 27, CH-8093 Zurich, Switzerland
chirkin@arch.ethz.ch

Reinhard König
ETH Zürich
Wolfgang-Pauli-Str. 27, CH-8093 Zurich, Switzerland
reinhard.koenig@arch.ethz.ch

ABSTRACT

This work presents a concept of interactive machine learning in a human design process. An urban design problem is viewed as a multiple-criteria optimization problem. The outlined feature of an urban design problem is the dependence of a design goal on a context of the problem. We model the design goal as a randomized fitness measure that depends on the context. In terms of multiple-criteria decision analysis (MCDA), the defined measure corresponds to a subjective expected utility of a user.

In the first stage of the proposed approach we let the algorithm explore a design space using clustering techniques. The second stage is an interactive design loop; the user makes a proposal, then the program optimizes it, gets the user's feedback and returns back the control over the application interface.

ACM Classification Keywords

D.2.2 Design Tools and Techniques: User interfaces; H.3.3 Information Search and Retrieval: Selection process; H.5.2 User Interfaces: Interaction styles; J.6 Computer-aided engineering: Computer-aided design (CAD)

Author Keywords

MCDM; interactive machine learning; urban design; multiple-criteria optimization

INTRODUCTION

Urban design decisions greatly affect the life of a city in many perspectives: the transportation network and the appearance are straightforward examples, but implications of the design go deeper into the citizens' experience. The effects of certain design decisions are not well-studied due to the complexity of the problem. Therefore, when working on urban design projects, it is common to decompose the problem into multiple aspects. Designers typically draw on past experience when subjectively prioritizing which aspects to consider with which degree of importance for their design concepts. The proposed project intends to aggregate the designers' past experience using data analysis techniques and optimization algorithms. This allows us to develop a planning support system that can help in the search for the best compromises for complex design problems.

The first challenge of a planning support system is formulating the problem: the designer's often vague qualitative requirements need to be translated into a precisely quantifiable criterion representation that can then be used in an optimization or generative algorithm. The second challenge is the fact that the priorities over the criteria depend on many factors varying with the context of the project and the designer's background: the set of good solutions may be too large and difficult to analyse, thus the program must model the context to narrow down the search domain.

This research presents a concept of interactive machine learning in an urban design context. The overall intention of the research is to improve task-related performance of the designers working with their software. Unfortunately, the nature of the application domain makes it difficult to evaluate the impact of a program on a designer's performance. One of the key performance factors in this area is the creativity of a designer. It has been argued that computer interface should be appealing, intelligent, and stimulating to endorse the creativity of an application's user[22] – thus an application is not allowed to disturb a designer focused on their work by asking too many questions in an active machine learning style. Avital and Te'Eni[2] build the concept of generativity which relates to the ability to create something new. According to Avital, two components of a task-related performance are the operational efficiency and the generative capacity; we aim at endorsing the generativity by proposing machine-generated design alternatives while trying keep the operational efficiency on a similar level with convenient CAD systems.

The goal of the research is to develop an algorithm that seamlessly integrates machine-generated design proposals into a human design process and is guided by a user's feedback. The core of the concept is a likelihood model of a designer's goals and preferences in a design session. The model is updated in a reinforcement learning loop using a human designer's feedback. The feedback comes in a binary form of design comparisons during a designer's work session. Using the maximum likelihood estimation (MLE) method in this model we construct a preference vector for a multiple-criteria design problem (MCDP) that arises in a human design. The preference vector allows reducing the design generation task to a

single-criteria optimization problem. Arranged into a continuous cycle with a user design session, a program lets a user and a machine iteratively work on a same design problem proposing alternatives and optimizing them towards changing user needs.

This paper describes an ongoing research. We overview the latest findings in adjacent disciplines, explain the model and an experiment setup. We present the intermediate results on a simplified test case; however, it is difficult to evaluate performance of the approach to this date.

BACKGROUND

The approach we propose relies on techniques from different fields of research. Although Avital and Te'Eni mainly focus on the generative fit of a program, referring to Frazer[7] and Janssen[11], they view artificial intelligence and other types of smart agents as a source for creativity[2]. Incorporating certain generative design (GD) algorithms, a program can inspire or challenge a designer by creating unique design alternatives[11]. Therefore, GD is one of the aspects we need to consider in our research. Singh and Gu [23] give a comprehensive overview of common GD methods, among which are: shape grammars, L-systems, cellular automata, swarm intelligence, and evolutionary algorithms.

Evaluation methods.

Evaluation of a solution (design) is an important part of design space exploration or optimization. The key concept within the scope of the paper is the design criteria that can be made explicit. Based on these criteria a user (designer) or a program can choose a preferable solution among available alternatives.

Quantifiable design criteria for urban design tasks include purely geometrical or topological measures, such as the length of roads or space accessibility[25], as well as social aspects, especially the perception of space, e.g. streetscape security[17]. We do not restrict the way the criteria are estimated; we state explicitly that the qualitative or subjective nature of some underlying aspects introduces an uncertainty into the evaluated criteria.

A popular group of methods for evaluation of an urban district form is called Space Syntax. It has first been conceived by Hillier and Hanson[10]. Space Syntax focuses on topological properties of a space like isovists (visible space from a point[4]), axial[25] or convex[19] open space. Besides special methods like Space Syntax, depending on a stage of planning or evaluation, one may use various direct statistical quantities: length of roads, area of recreational (and other) zones, amount of different types of facilities, etc. Some microclimate phenomena and their effects on the energy performance of buildings may be estimated using simulation methods. For example, high density urban areas feature increased temperatures due to the urban heat island effect[1]. Controlling a district morphology can help to mitigate this effect[21]. Surveys are used for evaluating social metrics of existing places. Salesses et al.[20] use high throughput internet surveys for evaluating perception of a city by the citizens.

Optimization methods.

Obviously, criteria formed by the evaluation methods are interdependent and sometimes contradictory. Thus, the designer faces a complex multiple-criteria design problem (MCDP) and wants to find the best compromises between the criteria. An approach to a MCDP that is widely used in parametric design is the exploration of Pareto-optimal solutions[27, 16]. The decision as to which of the Pareto-optimal solutions is best suited for a particular problem depends on qualitative criteria or non-operational human preferences. The concept of Pareto-front in GD is used in conjugation with evolutionary algorithms (EAs): these optimization algorithms allow a user to explore Pareto-optimal subsets of generated design proposals[11, 7, 14].

Although exploration of the Pareto-optimal solutions is a feasible approach to multiple criteria optimization, in urban planning and design it has two drawbacks: first, the Pareto front may be too large for being analyzed by a human; second, the desired solution might be far away from the optimal set in a solution space, because the designer may consciously sacrifice the optimality of some aspects for others. One way to find a desirable solution is to estimate the designer's priorities over the design criteria. This problem lies in the area of multiple-criteria decision making (MCDM) [13]. MCDM methods vary in a way they relate criteria to each other. The simplest approach is to make a single utility function as a linear combination of criteria; then the problem reduces to a search of weights (importance) for each criterion. This approach has a number of extensions that treat the weights, for example, as probabilities of being the most important criterion [18]. Many sociological studies argue that people tend to underestimate low probabilities [18, 13], thus more recent developments introduce uncertain methods and the fuzzy logic to utility models (e.g. [3]).

Data analysis.

There is no such a single measure to evaluate explicitly the quality of an urban district; and it is not clear how to asses the citizens' perception of a city in an absolute scale. Thus, one cannot ask a person to assess the quality directly. However, people are good at comparing and selecting: given a number of alternatives, a person could easily answer simple questions, like "where would you prefer to live?", or "which of these places looks more friendly?". They also can assign grades (labels, such as "good", "excellent", "bad", etc.), which then may be used in various learning-to-rank algorithms. The techniques of using these user assessments rapidly developed over the last decades due to the rising demand for them in data mining, information retrieval, and natural language processing[15, 26].

Salesses et al. [20] did crowd sourcing to gather pairwise comparisons of the images of four cities in the USA and Austria. They used the obtained data to score the streetscapes according to three different measures: class, safety, and uniqueness. This enabled them to correlate the scores with some measurable characteristics, such as income, population, or number of homicides. They found that the perception scores were able to

reflect the information about urban environment, which was not fully described by the income-based measures.

A recent research at MIT Media Lab [17] put further the ideas of P. Salesses. They showed the possibility to measure some aspects of human perception at high precision on a map[1] by using comparison data, image recognition, and machine learning techniques. The authors of the Streetscore algorithm used Salesses' dataset consisted of 208738 pairwise comparisons of streetscape images answering the question "Which place looks safer?". Then they ranked the images in the sample using the Microsoft TrueSkill algorithm [9] and evaluated the predictive power of various image features on the constructed score. Finally, they used the developed algorithm on a large number of images from Google Street View to make a high-precision map of the perceived street safety.

The Streetscore research is an example of a way to develop a design criterion that reflects social performance of an urban area. The key role in this approach is played by the TrueSkill scoring algorithm that allows constructing a rating of the elements in a data sample according to their pairwise comparisons. TrueSkill is a generalization of the Elo rating system; other modifications exist that have proven to be effective for scoring [24].

APPROACH

Interactive design process

The research does not aim at providing fully machine-generated urban design proposals. Instead, we want to develop a recommendation system that could be integrated into a design process conducted by a human. Figure 1 presents a UML diagram of the proposed machine learning and user interaction process. Process A shows the interaction:

- A.1 A designer creates the first version of a design.
- A.2 The program analyzes the design assuming it to be preferable for the designer. This allows making a hypothesis on the design goals.
- A.3 According to the created (machine) model of the designer's goals, the program suggests a small set of the machine-generated alternatives.
- A.4 The designer chooses one of the alternatives, thus giving additional information for refining the machine's model.
- A.5.1 The designer finishes the work, or continues to step A.1 creating a new design version.

On each iteration the designer submits a new design version; the program assumes it is better than a previous version – this gives more information for the machine's model of the designer's preferences.

The interaction cycle described above does not require providing any information besides the input it takes by observing a standard human design process: the only additional action the designer does is selecting the preferred solution among the proposed ones, which is itself the reason to use the application and the aim of the project. This setup can be viewed as a reinforcement learning model with human reward, which is a rapidly developing topic in machine learning (similar models are described in e.g. [5, 12]).

[1] http://streetscore.media.mit.edu/

Modelling data and features.

Let $X \in \Omega$ be a design descriptor - a random object in an arbitrary domain. In case of urban design X represents a single district layout, but it is not important in context of the described model. A (design) criterion is any numeric-valued function defined on a layout space. We assume aggregating output of this function into one or several values per layout. Then we consider m criteria $g_j(X) \in \mathbb{R}$. Or, the same: $\mathbf{g} : \Omega \to \mathbb{R}^m$.

Let $\mu_j = \mathrm{E} g_j(X)$, and $\sigma_j = \sqrt{\mathrm{Var} g_j(X)}$. Then define a set of normalized criteria by applying standard normal distribution function:

$$j \in 1..m, \qquad f_j(X) = \Phi\left(\frac{g_j(X) - \mu_j}{\sigma_j}\right), \qquad f_j(X) \in (0,1). \tag{1}$$

This gives a set of criteria functions that all lie in an interval $(0,1)$ and differ only in shape: $\mathbf{f} : \Omega \to (0,1)^m$, or in a shorter notation $\mathbf{f}(X^i) = \mathbf{f}^i \in (0,1)^m$. Given a data set containing n points, criteria values $\mathbf{f}$ become a matrix $F = \{f_{ij}\}_{ij}^{nm} \in (0,1)^{n \times m}$.

Next, we assume that a designer wants to optimize the layout according to the set of criteria. Hence, one implicitly has a desired value $y_j \in [0,1]$ for each criterion. Applying normal distribution function to the criteria allows treating in the same way the tasks of maximizing ($y_j = 1$), minimizing ($y_j = 0$), or converging to a particular value by setting appropriate y_j. A shorter notation is $\mathbf{y} \in [0,1]^m$.

By means of using the designer interaction described on Figure 1 process A, a designer provides the relational information in the form of designs and an answer to question (A.4): is one design better than another one according to the design requirements and the designer preferences?

We introduce a notion of an abstract quality of an urban design, that has no absolute measure, but rather is defined implicitly according to the relational data described above, and thus dependent on a particular designer and their design process. We combine the criteria into a single urban design quality measure using a preference weights vector. In MCDM this measure is referred as a (subjective) expected utility. Given a design goal – a vector of reference values $\mathbf{y}$, we use a goal programming utility function:

$$\theta(X) = \sum_{j=1}^{m} c_j (f_j(X) - y_j)^2. \tag{2}$$

Here c_j is a weight assigned to a criteria (i.e. preference). In a vector form: $\mathbf{c} \in [0,1]^m$. In order not to get degenerate fitness measure, we put constraints on the weights $\sum_{j=1}^{m} c_j = 1$. Given these constraints, the model has $2m-1$ degrees of freedom (m for $\mathbf{y}$ and $(m-1)$ for $\mathbf{c}$). Utility function $\theta(X)$ measures how far the layout X is from the designer's ideal.

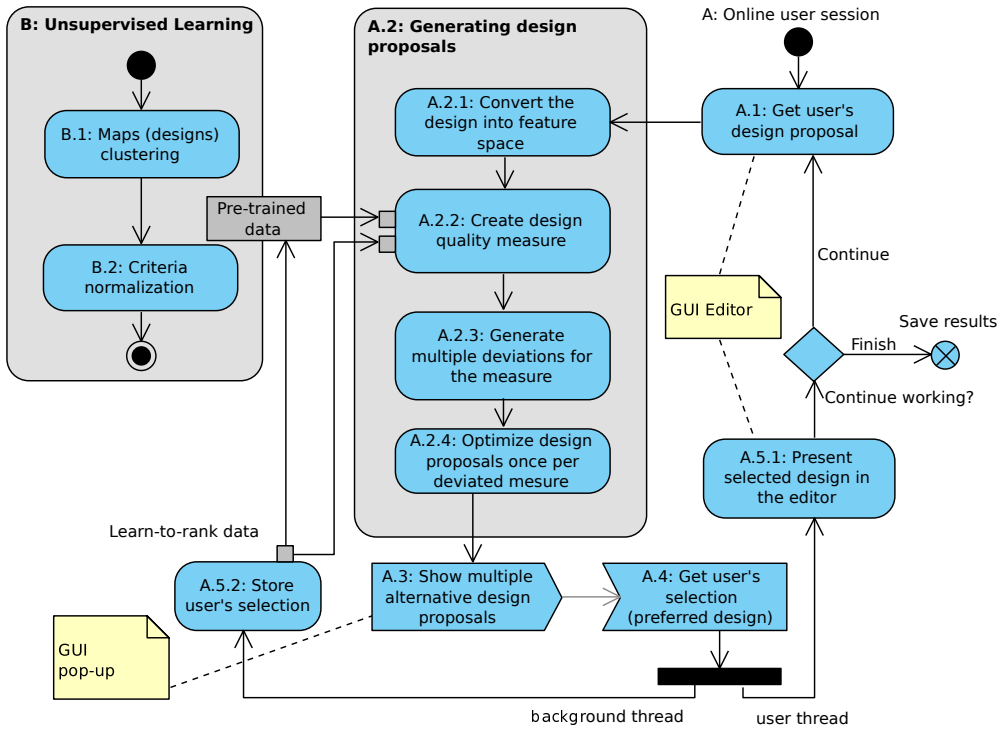


Figure 1: Learning cycle embedded into a design process

Because of the designer's feedback data, the preference vector helps to represent the quality measure rather as a reflection of the designer preferences (for a concrete design task) than as a fixed function. The creation of the preference vector for the design session is depicted as a step A.2.2 on Figure 1. Once the preference vector is known for a particular design type, the program can suggest the designer an alternative solution (A.3) by optimizing designer-created proposals according to certain quality measures (i.e. fitness function) (A.2.4). By modifying the preference vector, one can achieve the same effect as does the mutation procedure in genetic algorithms, hence introducing discrepancy into the possible solutions. This is to be done at step A.2.3 of the user interaction process. Note, the approach proposes to alter the fitness function instead of the generated solution; the solutions obtained by optimization according to different fitness functions are expected to vary, yet being optimal with respect to their measures.

Feedback

Given the problem statement, the information we can get from the user is *relative*, i.e. binary outcome for two layouts X^{i_1}, X^{i_2} whether one layout is better than another. In addition, user's feedback is highly subjective - a user may be uncertain whether one layout is better than another. We represent this uncertainty via random component - error, which results in a following model of layout performance:

$$p_i = -\theta(X^i) + \sqrt{2}s\xi_i, \qquad s > 0, \qquad \xi_i \sim \mathcal{N}(0,1). \quad (3)$$

Note the negative sign of θ: p_i represents performance of the layout – we model it as randomized negative of the bias 2.

Then the feedback of a user is represented as follows:

$$\delta_i = \begin{cases} -1 & p_{i_1} - p_{i_2} < 0, \\ 1 & p_{i_1} - p_{i_2} \geq 0. \end{cases} \quad (4)$$

Here $\delta_i = 1$ means that the user has chosen X^{i_1} and $\delta_i = -1$ means that the user has chosen X^{i_2}. δ is a random variable that is fully determined by random variables p_1 and p_2 (and by X if one models X as a random variable). Therefore, one can compute the distribution of the feedback δ_i:

$$\Pr(\delta_i = 1 | X^{i_1}, X^{i_2}) = \Pr\left(\frac{\theta(X^{i_2}) - \theta(X^{i_1})}{2s} \geq \xi_i \middle| X^{i_1}, X^{i_2} \right).$$

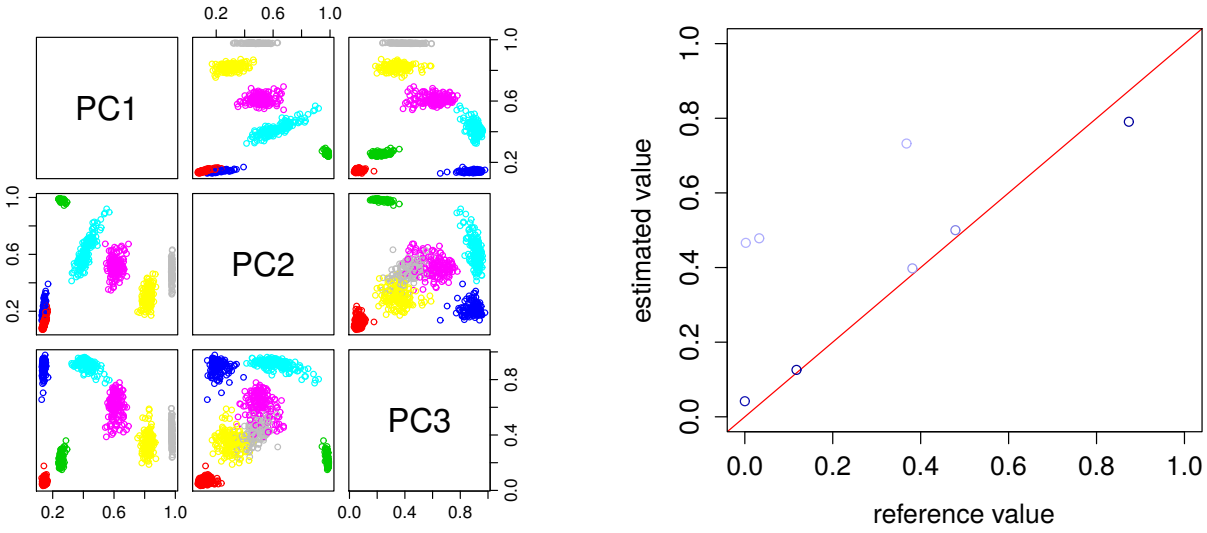
Let $a_{ij} = f_{i_2 j} - f_{i_1 j}$ and $b_{ij} = \frac{1}{2}(f_{i_1 j} + f_{i_2 j})$. Note, that in general they are strongly dependent as random variables and fully determined by X^{i_1}, X^{i_2}; $\{a_{ij}\} = A \in (-1,1)^{n \times m}$ and $\{b_{ij}\} = B \in (0,1)^{n \times m}$. Then the distribution of δ_i is defined as follows:

$$\Pr(\delta_i | a_{ij}, b_{ij}) = \Phi\left(\frac{\delta_i}{s} \sum_{j=1}^{m} c_j a_{ij} (b_{ij} - y_j) \right). \quad (5)$$

Matrices A and B, and vector δ are available in the dataset and $s, \mathbf{c}, \mathbf{y}$ are to be estimated. To simplify equation 5, we introduce a new variable $r_{ij} = \delta_i a_{ij}$, fully determined by the original variables; $\{r_{ij}\} = R \in (-1,1)^{n \times m}$. As a result, we get the final formula for a distribution of a user's decisioin δ_i:

$$\Pr(\delta_i | r_{ij}, b_{ij}) = \Phi\left(\frac{1}{s} \sum_{j=1}^{m} c_j r_{ij} (b_{ij} - y_j) \right). \quad (6)$$

Note, that $r_{ij} < 0$ when and only when the user feedback is "wrong", because then the utility difference $(\theta(X^{i_1}) - \theta(X^{i_2}))$



(a) Pair plots of the first three principal components of a point shape feature space. The points are coloured according to pre-defined shapes. The clusters are easily distinguishable.

(b) A biplot of estimating user preferences for one of the clusters. The more important (according to preference weights **c**) goals **y** are, the closer they are to reference values (the darker points).

Figure 2: Clusters of similar point shapes and a preference estimation for one of them.

and δ_i have different signs. A fraction of positive r_{ij} in available data may be a good measure of a problem difficulty.

Likelihood function.
Equation 6 allows estimating likelihood of the fitness measure parameters **c**, **y**, and model error parameter s: by adjusting these parameters one maximizes the span between $\theta(X^{i_1})$ and $\theta(X^{i_2})$, which increases the probability of getting "correct" δ_i. Now one can construct log-likelihood function:

$$\hat{l}(s,\mathbf{c},\mathbf{y}|R,B) = \frac{1}{n}\sum_{i=1}^{n} \log\left[\Phi\left(\frac{1}{s}\sum_{j=1}^{m} c_j r_{ij}(b_{ij}-y_j)\right)\right]. \quad (7)$$

Intuitively, equation 7 expresses the likelihood of parameters s, c_j, y_j given the dependence of the feedback δ_i on the sample a_{ij}, b_{ij}. By maximizing $\hat{l}$ one can find optimal values for parameters s, c_j, y_j. Considering $c_m = 1 - \sum_{j=1}^{m-1} c_j$, this model has $2m$ degrees of freedom, where m is a dimensionality of a criteria vector.

Learning model

Unsupervised phase
Process B on Figure 1 describes the unsupervised part of the machine learning process. As an initial dataset for the unsupervised learning we can use existing spatial configurations, which are freely available through OpenStreetMap. That is, we have an initial data sample $X_0 = x_1, ..x_{n_0}$ in a design space Ω, and a corresponding matrix of features (criteria) $F(X_0) = F_0 \in (0,1)^{n_0 \times m}$.

In the presented research we assume reference values **y** to implicitly depend on layout X. On unsupervised learning stage, however, we do not have an access to any designer's data to asses this dependence. Instead, we have a feature matrix F_0 that helps to infer a structure of the feature space: we use clustering techniques to group layouts X by their similarity in the feature space. For clustering layouts we use `R` implementation `mclust` of the expectation maximization algorithm by Fraley et al [6].

Classification
Unsupervised phase of the learning labels initial data, but after that any new data must be classified into one of the available clusters. This can be done by a variety of supervised learning methods; we use k-nearest neighbours algorithm implemented by Hechenbichler and Schliep in `R` package `kknn` [8], because it allows fast incremental classification during online phase, when new data points come one at a time.

Preference estimation
Once we have a label assigned to a particular design layout X_i, we can assume that design goal **y** does not change a lot within an assigned cluster. Thus we optimize a likelihood function 7 on a data subset from this cluster to estimate preference parameters of a designer in a given case. According to the proposed interaction scheme 3.1, the data that the algorithm gets on each iteration is a pair of layouts X^{i_1}, X^{i_2} and a decision feedback δ_i.

One problem in this setting is that we have to start with no data. To address this, we allow for training period, when the algorithm only analyses a user's actions. If there was a similar session (related to the same feature cluster), data from that session can act as a pre-training set.

Another problem is that layouts X^{i_1}, X^{i_2} might be on the "edge" between two clusters, or they, together with feedback δ_i, may contradict to other feedback in the cluster. These are addressed by adding a following heuristic to the classification phase: the data for preference estimation is taken only for those clusters,

preference directions for which are closer to the direction of a given layouts-feedback triple.

Layout optimization

The layout optimization is the final phase of the layout generation. The last layout X submitted by a user is optimized according to utility function 2, formed during the interactive design session, with a preference vector found by the previous phases of the algorithm. One can use convenient optimization algorithms for this phase. The major caveat here is that success of the optimization is highly dependent on the structure of the layout space Ω and its mapping on the features F.

SIMPLIFIED TEST CASE

The proposed concept is implemented on a toy design case of shaping a small set of points. A user is asked to form various patterns consisting of eight points, moving one point at a time using a mouse in a simple graphical interface. By pressing space bar on a keyboard, the user indicates design submissions, giving the program a necessary feedback. The feature (criteria) space for the problem is formed from all pairwise distances between points sorted in increasing order, plus additional statistical properties of a point set. After applying principal component decomposition, this feature set is invariant to points re-numeration, rotation, scale and shift.

Prior to the experiment with a human designer we create pre-training data for clustering by generating seven simple shapes with Gaussian noise and varying parameters: ellipse, parallel lines, rectangle, cross, T-shape, corner, single line. Figure 2a presents pair plots of three most significant principal components. Points on the figure are coloured according to the shape types. This figure shows that the constructed feature space is expressive enough to distinguish common layout types. Indeed, the clusters are visually separable.

Figure 2b presents the results of maximum likelihood estimation (maximizing function 7) applied on one of the clusters in the generated dataset (a user arranges the points in a rectangular-like shape). The plot compares the criteria values $\mathbf{f}_{\text{ref}}$ at a reference layout point X_{ref} (reference shape) to estimated goal values $\mathbf{y}$. The color intensity of a point y_i depicts the preference weight c_i of that point: the more importance the algorithm assigns to a point, the darker it is. Clearly, the optimization algorithm is better at estimating values of criteria that are more important due to the structure of the utility function 2.

DATA ACQUISITION AND SOFTWARE PROTOTYPE

In order to proceed with real design cases, we need to get data from close-to-real design problems. The main case study of a project is a reorganization of a small informal settlement in Cape Town. The level of details is restricted to a size of a single district and does not allow changes of building facades. A well-defined design problem on a given case study makes possible to list and discuss with designers their most important design considerations. Should the our approach prove its efficiency on the given case, it can be extended further to more general design problems.

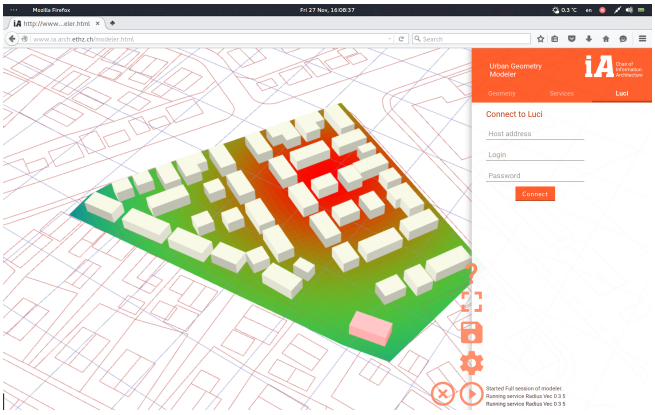


Figure 3: Qua-view is a front-end of qua-kit running in a browser.

To get enough data for the research we develop a simple web-tool called Quick Urban Analysis Kit (qua-kit) that is capable of editing geometry and visualizing computational analysis results. The tool is developed open-source at github.com[2]. Figure 3 presents a screenshot of qua-view – a front-end part of the tool running in a web browser. Qua-view is to be exposed to a wide audience, such as students of massive open online courses (MOOCs). The chair of Information Architecture at ETH Zürich develops several MOOCs on edX platform [3]. We create a set of exercises for students of these courses using the tool, so it serves two purposes: on the one hand, it provides an interactive learning environment for students, and, on the other hand, it gives us necessary feedback data to train the model and test the approach. The exercises are available online[4].

CONCLUSIONS AND FURTHER RESEARCH

The core ideas of the approach are the declaration of the data sources and the communication loop between a user and a program. We have developed the learning model based on changes to designs submitted by the user. This approach resembles a reinforcement learning model with human reward, which is a rapidly developing topic in machine learning (such models are described in e.g. [12]). We have also mentioned that the program may propose multiple design alternatives (Figure 1 A.3). Since the program can control generation of the alternatives, it can use active learning exploration-exploitation approach to improve its estimates. This reveals a lot of opportunities for further research.

A designer's priorities usually change during the design session as their proposal advances, hence a design session can also be modelled, for instance, as Markov decision process.

At the current stage of the project we are working on simplified geometries. Moving to real-world districts is a principle step towards completion of the project, and is to be done in near future.

[2] https://github.com/achirkin/qua-kit
[3] https://www.edx.org/xseries/future-cities
[4] https://qua-kit.ethz.ch/

ACKNOWLEDGEMENT

This work is partially supported by Scientific & Technological Cooperation Program Switzerland-Russia (STCPSR) 2015 (project IZLRZ1_164056).

REFERENCES

1. Jonas Allegrini, Viktor Dorer, and Jan Carmeliet. 2015. Influence of morphologies on the microclimate in urban neighbourhoods. *Journal of Wind Engineering and Industrial Aerodynamics* 144 (sep 2015), 108–117. DOI: http://dx.doi.org/10.1016/j.jweia.2015.03.024
2. Michel Avital and Dov Te'Eni. 2009. From generative fit to generative capacity: Exploring an emerging dimension of information systems design and task performance. In *Information Systems Journal*, Vol. 19. 345–367. DOI: http://dx.doi.org/10.1111/j.1365-2575.2007.00291.x
3. Liu Baoding and Chen Xiaowei. 2013. Uncertain Multiobjective Programming and Uncertain Goal Programming. *Journal of Uncertainty Analysis and Applications* (2013), 1–10.
4. M Batty. 2001. Exploring Isovist Fields: Space and Shape in Architectural and Urban Morphology. *Environment and Planning B Planning and Design* 28, 1 (2001), 123–150.
5. Wolfgang Ertel. 2012. Reinforcement learning combined with human feedback in continuous state and action spaces. In *2012 IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL)*. IEEE, 1–6. DOI: http://dx.doi.org/10.1109/DevLrn.2012.6400849
6. Chris Fraley, Adrian E Raftery, Thomas Brendan Murphy, and Luca Scrucca. 2012. mclust Version 4 for R: Normal Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation. (2012).
7. John Frazer. 2002. Creative Design and the Generative Evolutionary Paradigm. In *Creative Evolutionary Systems*. Elsevier, 253–274. DOI: http://dx.doi.org/10.1016/B978-155860673-9/50047-1
8. Klaus Hechenbichler and Klaus Schliep. 2004. Weighted k-Nearest-Neighbor Techniques and Ordinal Classification. *Discussion Paper 399, SFB 386, Ludwig-Maximilians University Munich* (2004).
9. Ralf Herbrich, Tom Minka, and Thore Graepel. 2007. TrueSkill: A Bayesian Skill Rating System. In *NIPS*. 569–576. DOI:http://dx.doi.org/10.2134/jeq2007.0177
10. Bill Hillier and Julienne Hanson. 1984. *The Social Logic of Space*. Cambridge University Press, Cambridge. DOI: http://dx.doi.org/10.1017/CBO9780511597237
11. Patrick Janssen. 2006. A generative evolutionary design method. *Digital Creativity* 17, 1 (2006), 49–63. DOI: http://dx.doi.org/10.1080/14626260600665736
12. W. Bradley Knox and Peter Stone. 2015. Framing reinforcement learning from human reward: Reward positivity, temporal discounting, episodicity, and performance. *Artificial Intelligence* 225 (aug 2015), 24–50. DOI: http://dx.doi.org/10.1016/j.artint.2015.03.009
13. Murat Köksalan, Jyrki Wallenius, and Stanley Zionts. 2013. An Early History of Multiple Criteria Decision Making. *Journal of Multi-Criteria Decision Analysis* 20, 1-2 (jan 2013), 87–94. DOI: http://dx.doi.org/10.1002/mcda.1481
14. Sivam Krish. 2011. A practical generative design method. *Computer-Aided Design* 43, 1 (jan 2011), 88–100. DOI: http://dx.doi.org/10.1016/j.cad.2010.09.009
15. Hang Li. 2011. Learning to Rank for Information Retrieval and Natural Language Processing. *Synthesis Lectures on Human Language Technologies* 4, 1 (apr 2011), 1–113. DOI: http://dx.doi.org/10.2200/S00348ED1V01Y201104HLT012
16. F Mallor, Pedro M Mateo Collazos, Isolina Alberto Moralejo, and Carmen Azcárate Giménez. 2003. Multiobjective evolutionary algorithms: Pareto rankings. In *VII Jornadas Zaragoza-Pau de Matem{á}tica Aplicada y estad{í}stica: Jaca (Huesca), 17-18 de septiembre de 2001*. Prensas Universitarias de Zaragoza, 27–36.
17. Nikhil Naik, Jade Philipoom, Ramesh Raskar, and Cesar Hidalgo. 2014. Streetscore - Predicting the Perceived Safety of One Million Streetscapes. In *CVPR Workshop on Web-scale Vision and Social Media*. IEEE, 7. DOI: http://dx.doi.org/10.1109/CVPRW.2014.121
18. Annibal Parracho Sant'Anna. 2015. *Probabilistic Composition of Preferences, Theory and Applications*. Springer International Publishing, Cham. DOI: http://dx.doi.org/10.1007/978-3-319-11277-0
19. John Peponis, Jean Wineman, Mahbub Rashid, S Hong Kim, and Sonit Bafna. 1997. On the description of shape and spatial configuration inside buildings: convex partitions and their local properties. *Environment and Planning B* 24 (1997), 761–782.
20. Philip Salesses, Katja Schechtner, and César A. Hidalgo. 2013. The collaborative image of the city: mapping the inequality of urban perception. *PLoS ONE* 8, 7 (jan 2013). DOI:http://dx.doi.org/10.1371/journal.pone.0068400
21. Saba Saneinejad, Peter Moonen, and Jan Carmeliet. 2014. Comparative assessment of various heat island mitigation measures. *Building and Environment* 73 (mar 2014), 162–170. DOI: http://dx.doi.org/10.1016/j.buildenv.2013.12.013
22. Ben Shneiderman. 2003. Leonardo's Laptop: Human Needs and the New Computing Technologies. (2003). DOI:http://dx.doi.org/10.1108/02640470310470598
23. Vishal Singh and Ning Gu. 2012. Towards an integrated generative design framework. *Design Studies* 33, 2 (mar 2012), 185–207. DOI: http://dx.doi.org/10.1016/j.destud.2011.06.001

24. Yannis Sismanis. 2010. How I won the "Chess Ratings - Elo vs the Rest of the World" Competition. (dec 2010). http://arxiv.org/abs/1012.4571

25. Alasdair Turner, Alan Penn, and Bill Hillier. 2005. An algorithmic definition of the axial map. *Environment and Planning B: Planning and Design* 32, 3 (2005), 425–444. DOI:http://dx.doi.org/10.1068/b31097

26. Ke Zhou, Gui-Rong Xue, Hongyuan Zha, and Yong Yu. 2008. Learning to rank with ties. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '08*. ACM Press, New York, New York, USA, 275. DOI:http://dx.doi.org/10.1145/1390334.1390382

27. E. Zitzler and L. Thiele. 1999. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation* 3, 4 (1999), 257–271. DOI: http://dx.doi.org/10.1109/4235.797969