CASE STUDIES ON THE INTEGRATION OF ALGORITHMIC DESIGN PROCESSES IN TRADITIONAL DESIGN WORKFLOWS

INÊS CAETANO[1], GUILHERME ILUNGA[2], CATARINA BELÉM[3], RITA AGUIAR[4], SOFIA FEIST[5], FRANCISCO BASTOS[6] and ANTÓNIO LEITÃO[7]
[1,2,3,4,5,7]*INESC-ID/ Instituto Superior Técnico, Universidade de Lisboa*
[1,2,3,4,5,7]*{ines.caetano|guilherme.ilunga|catarina.belem|rita.aguiar|sofia.feist|antonio.menezes.leitao}@tecnico.ulisboa.pt*
[6]*CiTUA/ Instituto Superior Técnico, Universidade de Lisboa*
[6]*francisco.bastos@tecnico.ulisboa.pt*

Abstract. Algorithmic design processes have enormous potential for architecture. Even though some large design offices have already incorporated such processes in their workflow, so far, these have not been seriously considered by the large majority of traditional small-scale studios. Nevertheless, as the integration of algorithmic techniques inside architectural studios does not require mastering programming skills, but rather taking advantage of a collaborative design process, small design studios are therefore able of using such strategies within their workflow. This paper discusses a series of challenges presented by one of these studios, where we had to integrate algorithmic design processes with the studio's traditional workflow.

Keywords. Collaborative design; Algorithmic design; Design strategies; Design workflow processes.

1. Introduction

The algorithmic revolution is changing the way architecture is practiced (Imbert et al. 2013; Heijden et al. 2015). This revolution was quickly adopted by several large-scale design studios, promoting a collaborative environment composed by multidisciplinary teams with different *know-how*. However, it has not yet spread to the majority of small-scale studios. Fortunately, this does not mean that they cannot benefit from the advantages of algorithmic approaches in productivity, cost/time reduction, and experimental freedom, among others benefits (Santos et al. 2012). To that end, it is important to combine traditional architectural design processes with modern, algorithmic-based ones. In this paper, we describe a series of practical case studies where this combination was followed, and we discuss the results, obstacles, advantages, disadvantages, and lessons learned from a collaborative design process involving a small-scale studio.

2. Algorithmic Design Integration and Collaborative Work

Collaboration in design is a practice introduced in the eighteen century, resulting from the divorce between the field of architecture and engineering (Giedion 1941),

T. Fukuda, W. Huang, P. Janssen, K. Crolla, S. Alhadidi (eds.), *Learning, Adapting and Prototyping, Proceedings of the 23rd International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2018*, Volume 1, 111-120.

prompting architects to work collaboratively with other experts. Nowadays, to face the emerging paradigms, such as algorithmic design, it became necessary to also collaborate with specialists with advanced programming skills.

A new reality of collaborative *architectural-engineering-mathematics* practice - which combines experts from different fields, including mathematics, robotics, and software engineering, among others - is present in some recent design teams, namely the Advanced Geometry Unit (AGU) of Arup, the Advanced Modelling Group (AMG) and the Computational Design & Research (CDR) of Aedas, the Specialist Modelling Group (SMG) of Foster and Partners, among others. These design studios demonstrate how, within the complex reality of parametric design and algorithmic techniques nowadays, collaborative work can promote a better and more efficient design process, in which the support of programming specialists is becoming increasingly important.

Our goal is to explore a collaborative design approach that allows small-scale design studios to take advantage of algorithmic processes. Previously, Caetano and Leitão (2017) explored the collaboration between two small-scale design studios and two algorithmic design specialists in the development of an algorithmic-based BIM façade for a residential building. In this paper, we follow a similar perspective, but we go much further: the collaboration also includes the algorithmic analysis and optimization of the algorithmically-generated design solutions.

In the next sections, we present three different examples developed collaboratively between a small-scale design studio and a small external team of algorithmic design specialists. The team's intervention was manifolded, focusing on geometrical, aesthetical, thermal, lighting, and structural issues. In the end, the case studies demonstrated that, even for small-scale studios and projects, the collaboration with algorithmic design experts can bring several advantages to the studio's design process.

3. Case Study 1 - Lighting Optimization

The first case study faced a complex situation of sun incidence control on a façade of an isolated private house facing the Atlantic sea. The aim of using algorithmic design was to generate a set of façade shading panels based on the concept of randomness that, at the same time, achieved a good daylight illumination performance.

The panels' geometrical pattern evolved considerably throughout the design process, resulting from the inherent capabilities of the algorithmic design approach, particularly, the quick visualization of the impact of changes, and the ability to simultaneously explore different conceptual directions. In the final design iteration, the architects decided on a geometrical pattern based on horizontal wood bars of different sizes, alternating between one full-length bar and a set of smaller bars (Figure 1.A). The size of the latter should be random, as also their position along the panel's width. To increase the architects' control over the panels' pattern, some restrictions on this random behaviour were set, namely the maximum and minimum sizes of the bars (*L-min* and *L-max*), the number of different possible sizes (*Δl*), and the maximum distance between bars (*D-max*) (Figure 1.B-C).

Initially, several design variations were produced by assigning different values to the design variables, allowing the architects to visualize and analyse a collection of results, and to then suggest improvements to be applied on the following iterations. As a result, this collaborative process of *generation-visualization-regeneration* helped the architects with the selection of the best values, from an aesthetic point of view, to be assigned to the design parameters.

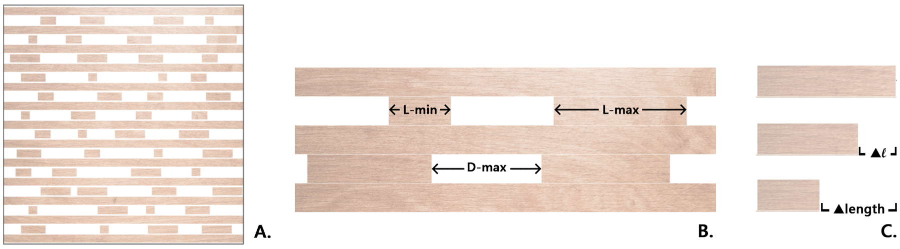

Figure 1. A conceptual representation of the shading panels' geometric pattern: A. alternation between bigger and smaller bars, which size and placement varies randomly. B. the pattern's geometric restrictions: bars' size (maximum and minimum length) and bars' maximum distance variables; C. size increment variable (the range of possible length sizes).

Since the architects' intent was to achieve a solution for the shading panels that was also optimized in terms of its lighting performance, in a second stage, we optimised the design regarding the *Spatial Useful Daylight Illumination* (sUDI) metric (Nabil and Mardaljevic 2006). To this end, we considered not only the set of design instances resulting from the architects' suggestions for the parameters, but also a larger design space that resulted from the assignment of values that deviated from those premises. Traditionally, this optimization would require manually changing the model and executing the corresponding analysis, repeating this time-consuming process until an acceptable solution was achieved. By using algorithmic processes, we can automate and speed-up this task.

We started by identifying the key rooms to be considered in the optimization process - i.e., the rooms where natural lighting was directly affected by the shading panels to be optimized - which helped significantly to reduce the time taken by each analysis. Then, different sampling techniques were used to generate different designs. Initially, Monte Carlo Sampling (MCS) (Shapiro 2003) was used, allowing us to test the optimization workflow. However, MCS requires a large number of samples to produce valuable results and, due to the costly evaluations required by lighting analysis, this is not time-efficient. To overcome this problem, Latin Hypercube Sampling (LHS) was then used to reduce the number of candidates, while improving the coverage and variance of the design space (McKay et al. 1979). In a first stage, it allowed us to obtain a design solution with 100% of *sUDI*, but with a high daylight glare probability (DGP), thus reducing the inside spaces' comfort.

The optimization process was then repeated, but this time considering the constraint set by the architects of using stripes of 5, 10, 15, 20, or 25 cm length. Only the distance between each stripe (the variable *D-max*) was dictating the light entering the room. We started by setting the *D-max* value as 20 cm, like the architects had suggested, and we generated 50 samples, obtaining a maximum *sUDI* value of around 45%, which was far from being optimum. Consequently, the optimization process was redone using, this time, a *D-max* value of 100 cm, generating 200 samples. The scatter plot in Figure 2 organizes the obtained sample, demonstrating that, until reaching a maximum distance of 50cm, the *sUDI* rapidly increases to 80%, whereas after that it slowly converges to 100%. Nevertheless, most of the solutions corresponding to higher *sUDI* values resulted from input values that deviated from the ones proposed by the architects.

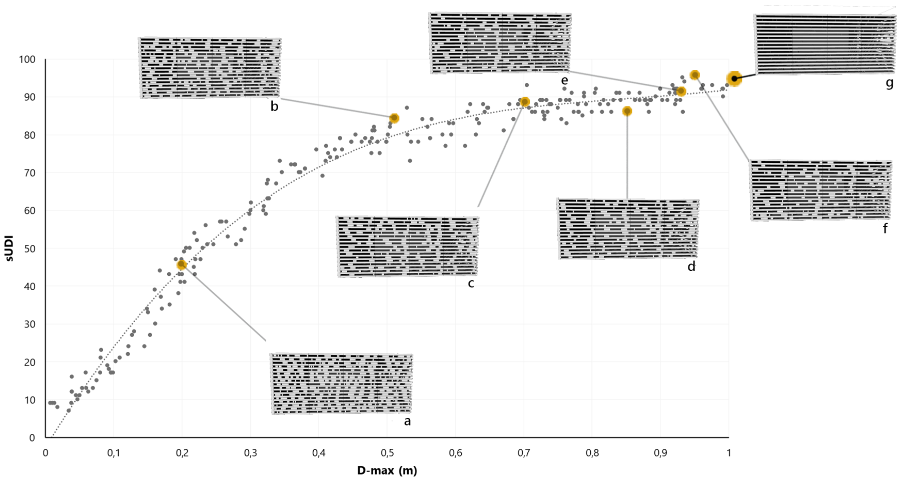

Figure 2. The scatter plot with the samples obtained during the optimization process. The models a. to g. correspond to the set of the examples presented to the architects.

The challenge at this stage was selecting a solution that not only had good lighting performance, but also matched the architects' design intent. To address this, we decided to evaluate how much an architect's initial suggestion restricts his final choice, i.e., the ease with which he accepts other design options that deviate from his initial idea. Therefore, we presented seven samples to the architects without informing them about the values of the variables and the corresponding *sUDI* levels. The samples were carefully selected in order to be heterogeneous (see Figure 2): option *a* fits all the constraints proposed by the architects; options *b-f* match all the constraints except the distance between bars (*D-max*), which increases from solution *b* to *f*; on the contrary, option *g* does not consider most of the constraints. After analysing the samples, the architects chose option *c* as the best option, option *d* as the second best option, and option *a* as the worst option.

In general, the options with a more balanced set of characteristics, i.e., with an acceptable *sUDI* value (higher than 80%) and, simultaneously, not deviating too much from the architects' original intent, were the ones that most pleased the architects. In contrast, none of the options with the highest *sUDI* values (options *e*, *f*, and *g*) were selected, which means that, regarding the architects' visual intent, these were not considered as good as the other options (options *b* and *c*). Nevertheless, neither were the former considered as the worst options, demonstrating that, even when the design deviates from the initial concept (option *g*), it may still be considered by the architect as a possible solution.

Regarding the worst option, even though it corresponded to the worst *sUDI* value, surprisingly, it was also the solution fitting all the constraints proposed by the design studio. The architects justified this choice as a result of the panels' pattern being excessively dense, suggesting that an additional design constraint should be added - the percentage of opening area should be at least 50%. This constraint can be easily implemented, and the whole analysis and optimization process can be automatically repeated, thus further illustrating the advantages of (1) the use of algorithmic design, analysis, and optimization, and (2) the collaboration between the members of the team and the design studio.

4. Case Study 2 - Structural Optimization

The second example considered is a two floor service building, located in an urban context, to work as a museum. The architects wanted to create an articulated truss structure to support the entire span of the building's roof slab. It was desired to minimize the depth of the truss solution adopted. To solve this problem, the structure had to be first designed to then be evaluated regarding the structural requirements of a service building. Using an algorithmic approach, the task of creating a model with the complexity of a truss structure is facilitated. Moreover, it allows us to easily produce different truss solutions, a fundamental step in the search for a truss design that meets the structural requirements. In this case, we selected the *maximum vertical displacement* as the metric to be studied and evaluated by using a structural analysis tool, in this case, *Robot* (Marsh 2014). We created an algorithm that generates different types of trusses to be tested, namely *Warren*, *Howe*, *Warren with verticals*, and *Pratt*, and, in this paper, we discuss the last two. We opted for spatial truss solutions where we can vary the number of the modular truss typology between supports, obtaining truss solutions with 6, 8, and 10 modules. Figure 3 shows a side and a plan view of a spatial *Warren with verticals* with 8 modules between supports, as well as a rendered image of the solution.

The different truss types were evaluated according to the regulation EN1993-1-1, adopted by European countries (CEN 2005). The national annex for the vertical deflections defines that, in the case of a multi-story service building, a limit of L/250 can be adopted, in which L is the smallest horizontal span of the building at the floor level of about 6,4 m. This is translated into a limit of 2,56 cm.

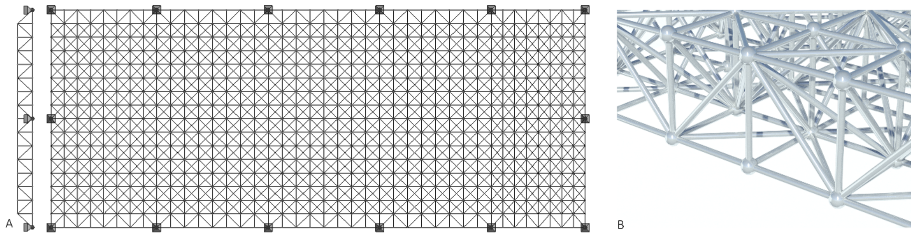

Figure 3. (A) shows a side and plan views of the spatial Warren with verticals truss type solution with 8 modules between supports, and (B) shows a rendered image of the truss type solution.

To automate the generation and evaluation of trusses, we used an iterative optimization process aiming at finding trusses with the minimum possible depth and the maximum displacement below the acceptable limit. Initially, the algorithm produced a set of samples covering a large design space of several depth values. Then, the algorithm narrowed the design space by focusing on a fraction of that space, which yielded the best results. It then produced the same number of samples, but, this time, on the sub-space. The algorithm continued searching until it reached a value close enough to the acceptable limit of maximum displacement. Similarly to Case Study 1, the algorithmic design team used MCS to test the optimization process' workflow at a first stage, later switching to LHS to better explore the design space. Figure 4 shows the maximum displacement results obtained.

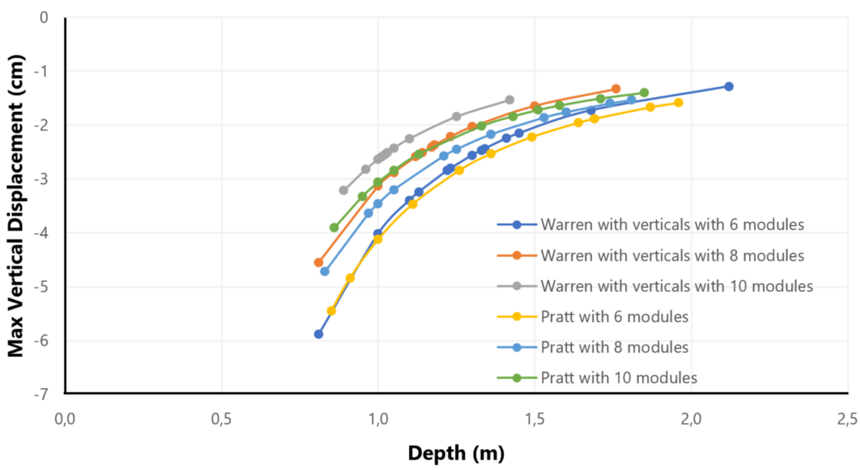

Figure 4. The displacements of the evaluated truss designs considering different depths and number of truss modules.

Observing the results, we can quickly conclude that the solutions with more truss modules have better performance in terms of deflection. The *Warren with verticals* solution with 10 modules achieves the smallest depth of 1.03 m with an acceptable maximum displacement of 2.51 cm.

The ability to assess the performance of the proposed solution allowed the architects to immediately acknowledge whether opting for a certain truss solution in the support of the building's roof is a feasible solution or not. As before, it was the capacity to automate the generation and analysis of several truss solutions that allowed architects to understand the impact of their design choices in the search for a better performing solution.

5. Case Study 3 - Automatic Furniture Layout

The third case study considers the generation of bedroom design configurations for a hotel being developed by the design studio. The project involved the rehabilitation and repurpose of an old XIX century residential building into a fully functional hotel with all the facilities and equipment required for this type of program.

One of the architects' challenges came from the necessity to furnish all the rooms with the necessary facilities and equipment to function as bedrooms. Given the building's original spatial organization, all the rooms had different configurations, shapes, and dimensions, as well as door and window positions. As such, the room equipment had to be specifically adapted to each room. In addition to that, the architects aimed at exploring variations for the placement of these elements. Unfortunately, given the large number of rooms to be furnished, manually adapting the room equipment to each room would have been a time-consuming task, limiting the exploration of different alternatives.

To address this problem, we developed an algorithm that generates different room solutions by placing the room equipment differently - i.e., bed, wardrobe, work table, shower, sink, and toilet. The algorithm considers as input the dimensions of the rooms, the positions of the door and windows, the dimensions of each element of the room equipment, as well as the circulation space needed to properly use them. With this information, the elements were stochastically placed and oriented within the boundaries of each room, while following a set of constraints that ensured the generated solutions would comply with both the constructability requirements and the architects' intent. The result of this implementation is a set of different design solutions for each room (Figure 5).

The algorithm evolved naturally from an open and constant dialogue between our team and the design studio. Initially, a set of solutions was generated and presented to the architects to be evaluated regarding their feasibility and suitability to the project. The feedback obtained allowed us to define new rules for the design, which were then translated into new constraints in the algorithm. This process of generation-visualization-regeneration was repeated iteratively, allowing us to shape the solutions based on the architects' feedback.

In addition to these constraints, the architects presented a set of preferences that the design should consider, e.g., the sink should preferably be placed next to the window. In practical terms, these preferences constitute soft constraints to the algorithm that, although not preventing the generation of solutions not fitting the preferences, are used to rank them. It is important to consider this type of constraints because, sometimes, the solutions that do not comply with the architects' preferences, are the only ones that are feasible.

This project is still ongoing: we are currently defining new constraints and preferences according to the architects' feedback in order to guide the algorithm towards better solutions.

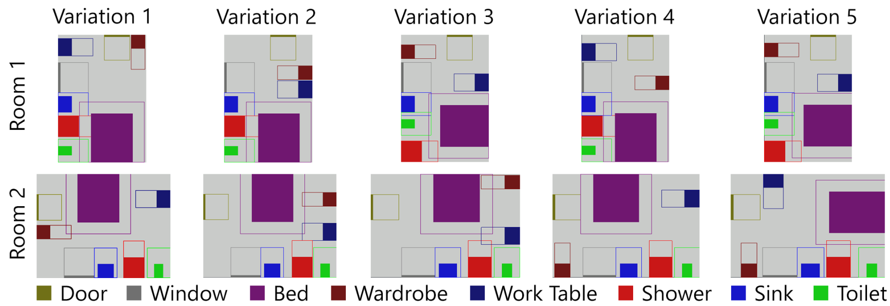

Figure 5. Solutions obtained for two different rooms. The filled rectangles represent the dimensions of the room elements, which were fixed by the designers, and the outlined rectangles represent the circulation space required for their use.

6. Conclusion

Despite the recognized advantages of algorithmic design techniques, traditional small-scale architectural offices still struggle to adopt them, mainly due to lack of time and resources. In these cases, the collaboration with digital design specialists and/or computation experts is a good alternative, which can bring the intended benefits without incurring dramatic changes in the studios' workflow and methods. In this paper, we discussed three different examples of such collaboration in which we were recently involved. Each example described a challenge that was presented to us by a traditional small-scale design studio.

Our goal was to use algorithmic methods to solve the studio's limitations and, then, to deliver a solution (or set of solutions) that could be used by them as a starting point for the following phases of their workflow. The challenges included (1) generating a set of façade shading panels with a degree of randomness and optimized regarding their lighting qualities, (2) creating a truss structure with the minimum acceptable depth for supporting the roof slab of a building, and (3) generating different possible solutions for furnishing a large number of hotel rooms.

In all cases, algorithmic-based design was used to produce the parametric versions of the intended designs. Then, sampling methods were used to drive the generation of design solutions until different fitness criteria were met. In all stages, a frequent interaction between the algorithmic design specialists and the design studio was critical: the generated solutions were presented and discussed with the architects, who then suggested additional constraints to be satisfied and preferences to be taken into account. This process was done through weekly meetings between the architects and the team of specialists, and it was repeated until optimal solutions, regarding their aesthetics and function, were achieved. The final solutions were then generated according to the architects' medium of choice so that they could simply import them into their work.

This approach allows the use of algorithmic-based design in a traditional/small-scale design studio without disturbing its working methods, thus representing an interesting solution for those studios that do not have the resources to create, in-house, their own algorithmic design team. Our approach follows the mixed-initiative proposal of Chaszar and Joyce (2016) of continuously involving the architects in the requirements elicitation and analysis, elaboration and implementation of prototype solutions, evaluation of the solutions by the combined team, and subsequent iterations of the process. The goal is for the studio to maintain control of the design process but, given the multitude of actors involved in the complexity of the tasks, it is inevitable that some control is lost (Chaszar 2016).

One important limitation of the process is that the algorithmic knowledge remains with the algorithmic design specialists. For large studios with sufficient resources, it is preferable to have that knowledge influencing the studio's practice. To that end, the studio should participate in the algorithmic developments with its own members, increasingly promoting in-house development (Sharples 2010).

Finally, in every work produced by a combined team, there is an important question that deserves to be discussed: who owns the intellectual property? Given the small scale of the projects we discussed, this was not a primary concern but we expect that, for larger projects, with considerably larger investments, it might become critical (Noble 2010). We do not have yet a solution for this problem but we hope our community will be a valuable source of ideas.

Acknowledgements

This work was supported by national funds through *Fundação para a Ciência e a Tecnologia* (FCT) with reference *UID/CEC/50021/2013*, and by the PhD grant under contract of University of Lisbon (UL), Instituto Superior Técnico (IST) and the research unit *Investigação e Inovação em Engenharia Civil para a Sustentabilidade* (CERIS).

References

Caetano, I. and Leitão, A.: 2017, Integration of an Algorithmic BIM Approach in a Traditional Architecture Studio., *Protocols, Flows and Glitches, Proceedings of the 22nd International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA)*, Souzhou, China, 633–642..

CEN, initials missing: 2005, *EN 1993-1-1: Eurocode 3: Design of steel structures - Part 1-1: General rules and rules for buildings*, European Comittee for Standardisation, Brussels.

Chaszar, A.: 2016, Issues of Control and Command in Digital Design and Architectural Computation, *CAADence in Architecture*.

Chaszar, A. and Joyce, S. C.: 2016, Generating freedom□: Questions of flexibility in digital design and architectural computation, *International Journal of Architectural Computing*, **14**(2), 167–181.

Giedion, S.: 1941, *Space, Time and Architecture*, The Harvard University Press, Cambridge.

Heijden, R.V.a.n.D.e.r., Levelle, E. and Riese, M.: 2015, Parametric Building Information Generation for Design and Construction, *ACADIA 2015: Computational Ecologies: Design in the Anthropocene [Proceedings of the 35th Annual Conference of the Association for Computer Aided Design in Architecture*, Cincinnati, 417–429.

Imbert, F., Frost, K.S., Fisher, A.W., Andrew Tourre, V. and Koren, B. 2013, Concurrent Geometric, Structural and Environmental Design: Louvre Abu Dhabi, *in* H. Lars, S. Sharma, J. Wallner, N. Baldassini, P. Bompas and J. Raynaud (eds.), *Advances in Architectural Geometry 2012*, Vienna: Springer Vienna, 77–90.

Marsh, K.: 2014, *Autodesk Robot Structural Analysis Professional 2015: Essentials*, Marsh API LLC.

McKay, M.D., Beckman, R.J. and Conover, W.J.: 1979, A Comparison of Three Methods for Selecting Value of Input Variables in the Analysis of Output from a Computer Code, *Technometrics*, **21**(2), 239–245.

Nabil, A. and Mardaljevic, J.: 2006, Useful daylight illuminances: A replacement for daylight factors., *Energy and buildings*, **38**(7), 905–913.

Noble, C. 2010, Controlling Intellectual Property, *in* P. Deamer and P.G. Bernstein (eds.), *Building (in) the Future: Recasting Labor in Architecture*, Princeton Architectural Press, New York, NY, USA, 171-177.

Santos, L., Lopes, J. and Leitão, A.: 2012, Collaborative Digital Design: When the architect meets the software engineer., *H. Achten et al., eds. Digital Physicality - Proceedings of the 30th eCAADe Conference.*, Czech Technical University in Prague, Faculty of Architecture (Czech Republic), 87–96.

Shapiro, A. 2003, Monte Carlo Sampling Methods, *in* A. Ruszczynski and A. Shapiro (eds.), *Handbooks in Operations Research and Management Science*, Elsevier Science, 353–425.

Sharples, C. 2010, Technology and Labor, *in* P. Deamer and P.G. Bernstein (eds.), *Building (in) the Future: Recasting Labor in Architecture*, Princeton Architectural Press, New York, NY, USA, 90–99.