An optimization strategy for 3D concrete printing

An Optimization Strategy for 3D Concrete Printing

Rob Wolfs, r.j.m.wolfs@tue.nl
Eindhoven University of Technology, The Netherlands

Theo Salet, t.a.m.salet@tue.nl
Eindhoven University of Technology, The Netherlands

Abstract
The development of automation techniques shifts the building industry towards CAD/CAM oriented design. This requires the relations between design and construction to be considered in each step of the design process. However, the nature of these relations is still largely unknown for new techniques and the amount of involved parameters is great. One of these new, upcoming techniques is 3D printing. A research model has been developed, which allows 3D concrete printing to be studied in an efficient way. The model is set up in a parametric environment, combining a structural analysis with an optimization module based on the simulated annealing algorithm.

Keywords: CAD/CAM, 3D Concrete Printing, Optimization, Simulated Annealing

1 Introduction

The on-going developments of automation techniques in the building industry require each discipline in the construction cycle to change their approach. Design (CAD) and construction (CAM) are no longer separate, but must be considered as connected from early sketching to realization. One of these new automation techniques is 3D printing. In case of 3D concrete printing, the connection consists of the relationship between the different components involved: printing strategy, the printed material (concrete) and the printed objects. However, the nature of these relations is generally unknown, which may result in a time consuming trial-and error approach to find the desired printing strategy if one or multiple parameters change. A graduation project on 3D printing of concrete structures has been carried out at the Department of the Built Environment of the Eindhoven University of Technology. The project aimed at contributing to the development of the technique, by designing a method to study and optimize the relationships between the components involved in 3D concrete printing in an efficient way. The topics discussed here are based on the findings of this project (Wolfs 2015).

2 Research Model

A research model has been constructed to study the concrete printing technique without the need for a trial-and-error approach. The structure of this model is depicted in **Figure 1**.

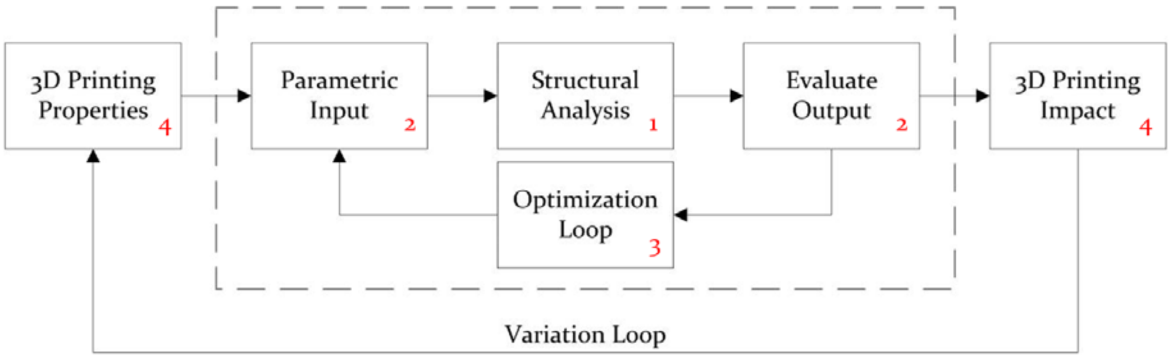

Figure 1 Research Model

The core of the research model is a structural analysis module [1], realized in FEM software Abaqus. This allows the structural behaviour of printed objects to be studied for varying shapes, as it includes orthotropic behaviour which is a typical result of the layered production method. Considering the developments of new printable materials like fibre-reinforced concrete, non-linear elastic material properties are also incorporated in the FEM model.

The input and output modules [2] are set up in a parametric environment, which provides the user with the possibility to easily vary parameters and their relations. The input can be categorized in geometry, boundary conditions, and material parameter sets. All of these are modelled in Grasshopper, a graphical algorithm editor which comes as a plugin for Rhinoceros.

The great number of connected parameters results in a large amount of combinations and variations. To avoid the need for an exhaustive search on desirable combinations of settings, an optimization module is incorporated in the research model [3]. This module uses the output of the structural analyses to find the desired input settings.

In addition to the parameters in the input component, a set of printer properties is added to the model. They consists of a user defined printing strategy, i.e. printing speed, layer size and printing environment. These properties are placed in a variation loop. By changing the print strategy in a stepwise manner, entering the optimization loop for each step, the impact of 3D printing can be studied [4].

As Grasshopper is a propagation-based system, it restricts the use of cyclic algorithms (i.e. 'loops') without the use of additional plugins. To keep full control of both the optimization and variation loop, these modules are written in programming language Python. Additionally, Python can be used to control the structural analysis in Abaqus.

3 Optimization Algorithm

Because of the early stage of research 3D concrete printing is in, little is known about the behaviour of the components involved and their relations. When evaluating one or multiple printing parameters and searching for a certain goal, it is initially unknown how the range of the parameters' values is related to the optimum value. Moreover, the evaluation function, i.e. the way the chosen values relate to the optimum (the fitness of each solution), is usually strongly non-linear and may contain local optima. Fortunately, optimization algorithms have been improved over the past years. Methods have been developed that take these issues into account, like the simulated annealing (SA) algorithm.

3.1 Simulated Annealing

The simulated annealing algorithm was inspired from thermodynamics and metal work. Annealing involves heating and cooling a material, altering its properties by changing its structure on molecular level. Once the material is cooled down, this new structure is locked in, along with its newly obtained properties. If the temperature is dropped too quickly, irregularities may occur which are trapped in the newly created structure as well (Jacobson 2013).

The SA algorithm has a fictitious temperature variable, similar to this heating and cooling process. This variable starts high, and the slowly 'cools down' as the algorithm runs. At high temperature the algorithm is often allowed to accept worse solutions than the currently best one, more or less similar to a random search. It is therefore able to jump out of local optimums in the early stage of execution, like depicted in **Figure 2**. The chance of accepting worse solutions is reduced as the temperature drops, i.e. as the algorithm has completed more iterations. This allows the algorithm to narrow the search space. (Michalewizc & Fogel 2000).

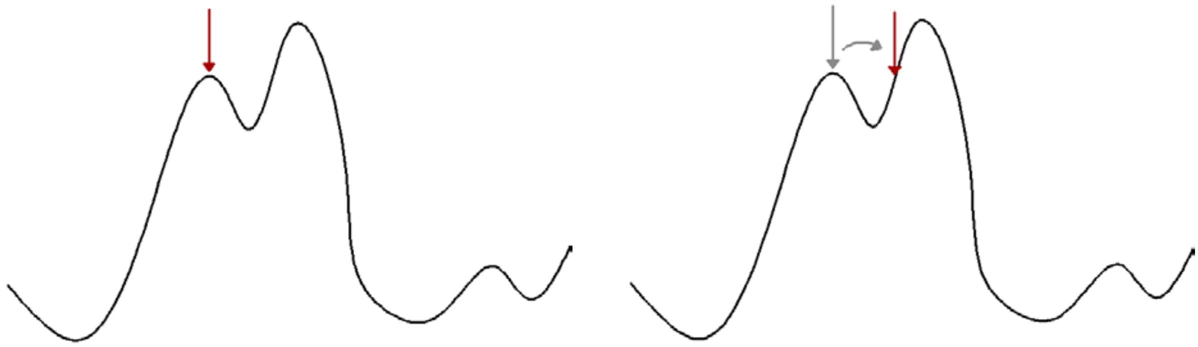

Figure 2 – Example of a simulated annealing algorithm jumping out of a local optimum

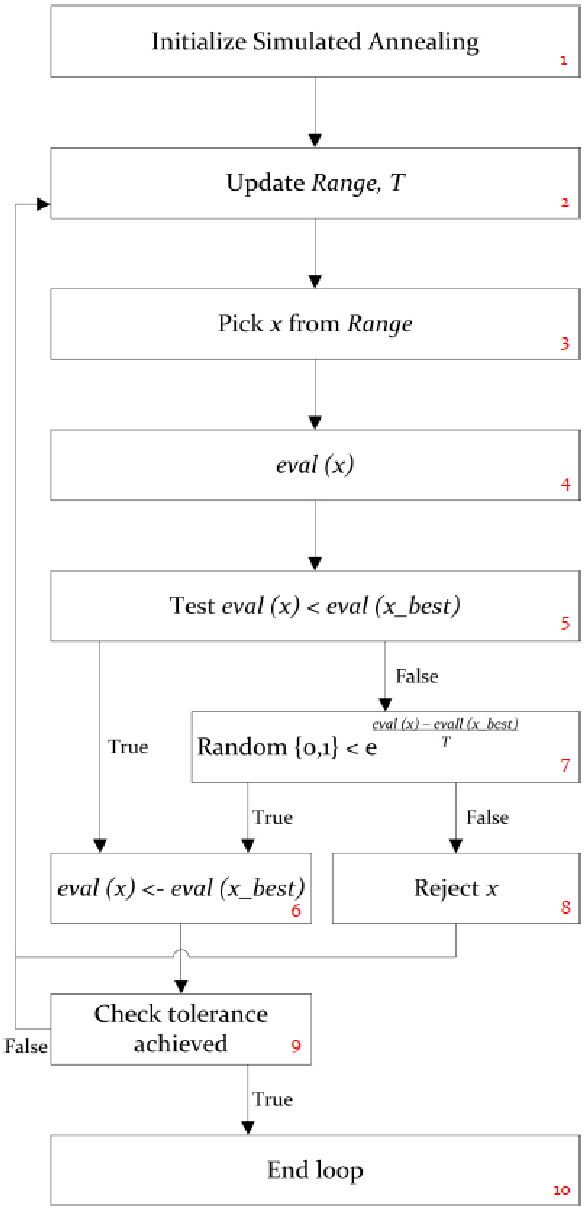

Figure 3 – Flowchart of simulated annealing as used in the research model

3.2 Implementation in Research Model

The optimization module of the research model is based on the SA algorithm. The procedure is given in **Figure 3**. Each function of the algorithm will be explained stepwise based on this flowchart.

As the simulated annealing algorithm gets initiated [1], the variable and goal of the optimization loop have to be fixed, along with a certain number of parameters. The target module is parametrized in Grasshopper and allows the users to choose these parameters of interest: the variable to be optimized can be the applied loading, the printed geometry, or material properties. The chosen optimization goal is based on the structural behaviour of the printed object, and can be either the occurring stresses or displacements.

Next, the algorithm parameters are chosen. The maximum amount of iterations is fixed, and so is the initial domain and temperature [2]. Finally, the tolerance is set. Once the algorithm has found a solution that is within this range of accuracy, it terminates the loop.

The algorithm is now initialized and choses a random point *x* within initial domain {*R1, R2*} [*3*]. The value *x* is then evaluated by the function *eval(x)*, which corresponds to a structural analysis in Abaqus [*4*]. This analysis will show the maximum occurring stresses or deformations and compares them with the predefined goal. The result of this evaluation function is compared to the currently best solution [*5*]. If the occurring result is indeed better, *eval(x)* becomes the currently best solution *eval(x_best)* [*6*]. When *eval(x) < eval(x_best)* returns a *False* statement, a new function is called upon. This part of the algorithm determines if a worse solution is accepted or not, based on the difference between the current and best solution, and the fictitious temperature parameter *T*. The probability of acceptance is chosen as a random value in the domain {0, 1} [7]. This number is compared with the evaluated point in Python as follows:

```
p = math.exp((eval(x) – eval(x_best))/ T)
if random.random() < p:
    eval(x_best) = eval(x)
```

Once this statement returns *True*, the value is accepted as *eval(x_best)* [*6*]. When it returns *False*, the solution is discarded [8]. After these steps, the solution being either better or worse, the algorithm will return to its starting point and repeat the same steps. Before doing so however, it checks a termination condition; in this case the required tolerance of the solution [*9*]. If the tolerance is achieved, the algorithm ends, showing the result, the corresponding best value of *x* and the tolerance percentage of the solution [*10*].

If the tolerance is not achieved, the temperature *T* is updated before restarting the algorithm. The nature of this update function can be varied: both the order of the function and the size of steps may vary between analyses. Subsequently, the choice of function strongly influences the probability of accepting worse solutions. In case the analysed problem is largely unknown or contains a high amount of local optima, it may be desired to slowly decrease the temperature during the run. On the other hand, when the problem is well-known, the temperature function may drop rapidly such that the algorithm converges quickly to the optimum solution. For the implementation in this research model, the temperature update function is taken as linear decreasing. Each step *k* in temperature drop is therefore equal, but the step size can be varied by selecting the total number of iterations *I*. The resulting temperature function in Python language is shown below.

```
def update_temperature(T, k):
    return T_int – (k * (T_int / I) )
```

To study the concrete printing technique an additional functionally has been added to the algorithm. The behaviour of the parameters of interest and the relationship between them is generally unknown. For this reason, the initial domain {*R1, R2*} is taken very large at first, to prevent a too small choice of range that doesn't incorporate the optimum solution. However, as the algorithm runs, the problem gradually becomes clearer. Keeping the domain large would result in a time consuming optimization run, which negates the benefit of this smart algorithm. Thus, at a user defined point in the loop, the algorithm will narrow the initial domain to {R3, R4} step by step based on the current best solutions. This allows the algorithm to converge to an optimum solution much faster, even when the problem is initially unknown.

The values of R3 and R4 are based on the best and second best known solution, which continuously update during the run, as the figures below illustrate. The horizontal axis concerns the domain, in which a parameter *x* may be chosen. The vertical axis lists the fitness of each solution *eval(x)*. The optimum solution corresponds with an unknown optimum variable *x_optimum:* the target of the algorithm.

Figure 4 shows that if this expected optimum is somewhere in between the currently best two solutions *x1* and *x2*, i.e. the best evaluation value is below the target and the second best is above or vice versa, the range will be straightforwardly reduced.

However, if the initial range was chosen too small and both values are on one side of the target, the new range is less obvious. **Figure 5** shows how the algorithm uses the distances between the two best values *x1* and *x2*, to compute the trend of the solutions. This can then be used to estimate the distance Δx between the currently best value and the optimum one. However, as this assumes the problem to be linear, the range is taken as twice this distance $2\Delta x$ to incorporate for non-linear optimization problems.

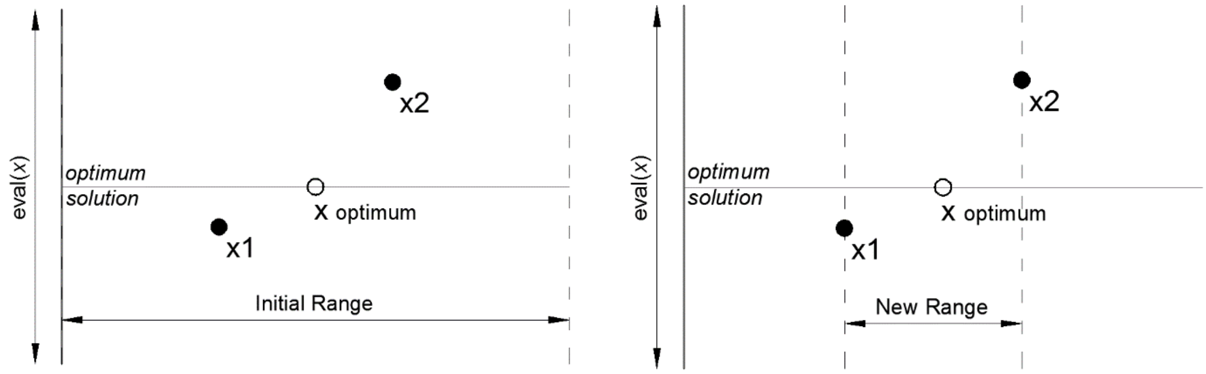

Figure 4 – Reduction of range, using the two best known evaluation values on two sides of the optimum

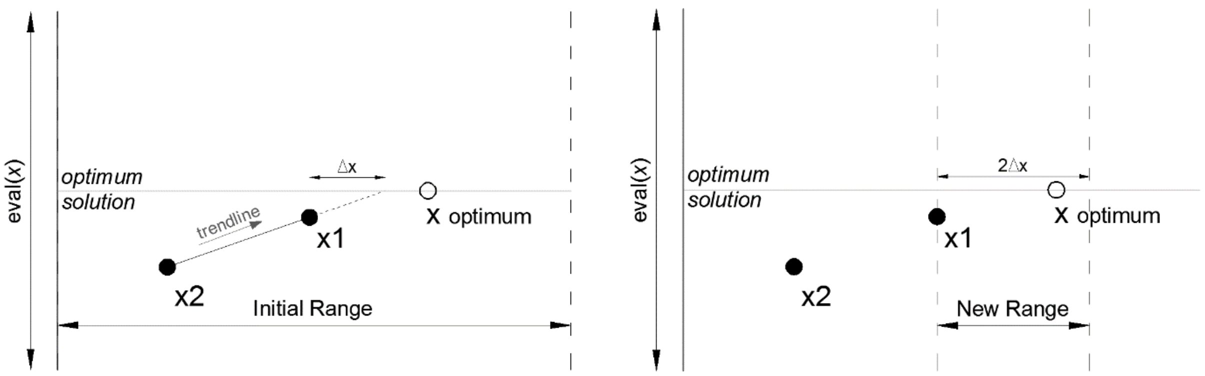

Figure 5 – Reduction of range, using the two best known evaluation values on one side of the optimum

It is noted that when using optimization algorithms, the balance between computational time and accuracy comes into play. Besides, accepting solutions is partially based on chance. Especially in case of largely unknown problems, as is this case for this study, either much iterations and exact solutions, or a quick evaluation and somewhat inaccurate solutions may be the result. For this early stage of research the latter is preferred.

4 Example

The research model is demonstrated by varying the printing strategy of a 3D printed orthogonal concrete wall. The printing speed, layer size and printing environment (in-situ or pefab) are varied in a stepwise manner. For each step, the optimization loop is entered, which seeks toward the maximum loading capacity of the wall. As the optimization loop runs, the results of each step are sent back to Grasshopper in real time, allowing the user to keep track of the progress. Finally, the optimized result in presented, which is done for each step in the variation loop. These results can be combined and presented in a graphical way, to show the impact of 3D printing on the design process. An example is given in **Figure 6**. This graph shows the normalized printing speed on the horizontal axis, versus the normalized loading capacity on the vertical axis. Each dot represents the result of an optimization loop.

5 Conclusions

The example analyses of the model show that the impact of 3D printing should not be underestimated, as the chosen printing strategy clearly influences the required mechanical properties or geometry, and vice-versa. The presented research model provides an efficient way to study 3D concrete printing, without the need for a time-consuming trial and error approach. The modular and parametric environment of the research model are set up such, that they can easily be replaced or edited by fresh data on existing or newly developed print techniques, as the following years will provide much more (experimental) research on 3D concrete printing.

It is recommended that future research will be carried out to extend and renew models like presented here, to support upcoming automation techniques like 3D concrete printing and apply them on a large scale in the building industry.

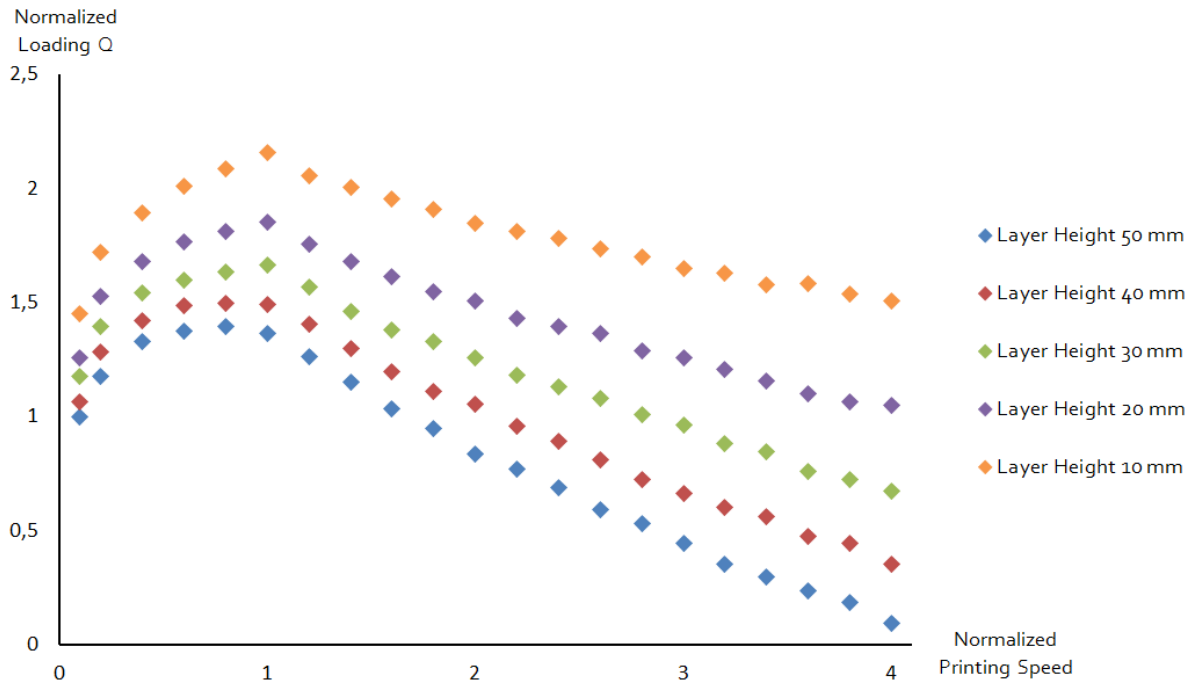

Figure 6 – Optimization results of printing speed versus loading capacity for varying layer height

References

Jacobson, L. (2013). *Simulated Annealing for beginners.* Retrieved January 2015, from theprojectspot.com.
Michalewicz, Z. & Fogel, D. (2000) *How to Solve It: Modern Heuristics.* Berlin: Springer.
Wolfs, R (2015). *3D Printing of Concrete Structures.* Master's thesis. Eindhoven University of Technology.