ABSTRACT
Ruin has a bad name. Despite the obvious complications, failure provides a rich opportunity—how better to understand a building’s physicality than to watch it collapse? This paper offers a novel method to exploit failure through physical simulation and iterative machine learning. Using technology traditionally relegated to special effects, we can now understand collapse on a granular level: since modern-day physics engines track object-object collisions, they enable a close reading of the spatial preferences that underpin ruin. In the case of bricks, that preference is relatively simple—to fall.

By idealizing bricks as rigid bodies, one can understand the effects of gravitational force on each individual brick in a masonry structure. These structures are sometimes able to ‘settle,’ resulting in a stable equilibrium state; in many cases, it means that they will simply collapse. Analyzing ruin in this way is informative, to be sure, but it proves most useful when applied in series. The evolutionary solver described in this paper closely monitors the performance of constituent bricks and ensures that the most successful structures are emulated by later generations. The tool consists of two parts: a user interface for design and the solver itself. Once the architect produces a potential design, the solver performs an evolutionary optimization; after a few hundred iterations, the end result is a structurally sound version of the unstable original. It is hoped that this hybrid of top-down and bottom-up design strategies offers an architecture that is ultimately strengthened by its contingencies.
INTRODUCTION

Bricks don’t aspire to be in arches—they want to be on the ground. As a kind of fundamental antagonist of architecture, gravity is sadly under-represented in the digital realm: consider the surfaces that weightlessly float across your screen each day. What’s needed now is a kind of realism about materiality, an acknowledgement that buildings can and do fall down.

Though the losses associated with failure can often be devastating, the ruined building sometimes provides an opportunity for realignment: look no further than the cathedral at Beauvais. Intended as a structurally audacious successor to the cathedral at Chartres, the ambitious 144 ft vaulting collapsed in 1284, causing later additions to be scaled down significantly. Contemporary builders certainly took note; Grodecki (1976) characterizes the collapse as “signaling the end of the gigantic Gothic undertakings of the thirteenth century.” As it stands, the cathedral has been the site of ongoing structural modification, playing host to a variety of ad-hoc supports in an effort to keep the daring structure together.

Architecture’s recent symbiosis with structural engineering now largely mitigates our risks of collapse. A deep understanding of a building’s statics is now de rigueur; while this development has certainly resulted in safer buildings, the opportunity to learn from structural failure is almost entirely absent from contemporary discourse. Building to failure allows for limit states to be reached and for analysis to move beyond the statically indeterminate. In short, a ruined building offers a richer diagram than that achieved by analysis alone (Figure 2).

This new approach aims to both exploit and iterate that diagram. Recent developments in rigid-body physics engines allow us to simulate the spatial preferences of vast numbers of independent bodies; the collapse of an entire masonry building can be simulated over the course of a few minutes. New analysis tools can perform a close reading of collapse and provide a detailed understanding of each brick’s stability over time.

The use of rigid-body physics in architecture is hardly without precedent (Dierichs and Menges 2013; Dierichs and Menges 2012; Meredith and Sample 2013). Though previous studies have focused on the simulation of relatively unstructured aggregates, the application of rigid-body physics to traditional masonry construction has pedigree in the world of structural engineering. Jacques Heyman has argued for an emphasis on stability over strength in the analysis of structural masonry, noting that the maximum stresses encountered in a typical dry-stone cathedral are insignificant compared to the crushing strength of the material. The cathedral at Beauvais, for instance,
was found by Benouville to have a maximum stress of only 1.3 N/mm$^2$, only 1/30th of the stone’s compressive strength of 40 N/mm$^2$ (Heyman 1995). Clearly, the structural issues at play were failures of stability; in most dry masonry applications, abstracting each brick as a rigid body should accurately model the failure mechanism at play.

This approach stands in contrast to the pioneering computational masonry work undertaken by the Block Research Group, which adapts the traditional graphical method of funicular design to a three-dimensional interactive design tool. Not unlike the hanging-chain method employed by Gaudi, Otto, and others, the success of this approach lies in its abstraction: funicular vaults are accurately modeled through ‘Thrust Network Analysis,’ a form-finding method that models the structures as ‘spatial representations of compression forces in equilibrium with the applied loads’ (Rippmann and Block 2012).

The algorithm described in this paper is modeled on a considerably less elegant method: the centuries-long process of trial-and-error structural optimization. As Heyman notes, the historical evolution of masonry was led mostly by the study of precedent—in the absence of static analysis, builders were encouraged to follow rules of thumb and expand on passed-down experience. The machine-learning techniques described herein fulfill a similar purpose, as successful iterations are copied and mutated until new, better precedents are established. By automating the process of collapse and analysis, this new algorithm effectively short-circuits the historical ‘brute force’ approach to establish a novel general method for masonry structural optimization.

**METHOD**

Before the evolutionary solver begins its iterative collapse, a user interface allows the designer to specify a pre-simulation genotype, encoding a template within which the physics engine can begin to place its bricks. After simulation occurs, the evolutionary solver mutates the initial genotype and passes the resulting template back to the physics engine; the process is then iterated and successful mutations are stored within the evolved genotype.

Beyond simply controlling the tower’s shape, the user interface also allows for bricks to be removed from its overall form. This has the effect of both increasing the solver’s rate of attrition and making the resultant collapse less computationally expensive. The first version of the solver removed these bricks using cellular automata rules (Figure 3) (Wolfram 2002), encoding a degree of spatial preference in each brick and resulting in a truly object-oriented outcome. Later versions of the solver implemented a top-down approach, controlling on/off states with an 18-parameter sinusoidal equation.

The net effect of both methods is that of aesthetic porosity. Although the cellular automata technique was conceptually preferable, initial results showed that slight variations in the solver’s Wolfram rule genotype had systemic rather than incremental effects on the phenotype. Further, the equation method allows deep customization and a fine-grained approach to incremental
change that dramatically increases the performance of the solver.

Computational efficiency lies behind the simulation strategy as well. Vaulted masonry structures have traditionally relied on temporary centering for support during construction. Early versions of this algorithm followed suit: instead of resting on wooden scaffolding, the bricks were allowed to propagate before the physics engine was enabled. This created a kind of bottleneck, as each simulation started with every brick in play. In order to ease the computational load, later versions of the algorithm adopted a brick-by-brick approach; after each course is laid, the physics engine is enabled for long enough to detect and remove any unstable bricks.

Aside from the obvious correlation with physical bricklaying, this approach has historical precedent: Brunelleschi’s dome in Florence was famously constructed without interior centering, though this meant that the dome needed to be far sturdier than those supported in the traditional way (King 2000). This approach may be stricter than real-life construction, but it stands to reason that an incrementally simulated masonry structure will be easier to construct than one conceived with virtual ‘centering.’

A rigid-body physics engine known as Bullet performs the simulation itself. Originally written for C++ by Erwin Coumans, it was later ported to Java by Martin Dvorak; the current implementation of my own algorithm uses bRigid, an excellent Processing.org port written by Daniel Kohler. This engine works in close conjunction with the evolutionary solver, which was written from scratch using typical strategies for evolutionary computation (Dillenburger, Braach, and Hovestadt 2009; Rosenman 1996; von Buelow 2002). The genotype of the solver is an array of eighteen parameters which control both the aforementioned sinusoidal equation and the initial rotation and separation of each brick (Figure 6). The fitness criterion is a user-weighted factor of brick placement success (i.e., the percentage of intended bricks that remain standing) and height, divided by the overall number of bricks required to reach that height. The obvious criteria here might be the total height divided by number of bricks standing, but this almost always results in the generation of a single-brick-width tower. Factoring in the brick placement success rate encourages the solver to build complete courses and enforces the “design intent” of the sinusoidal equations.

A single parameter of the genotype is mutated for each iteration; as each course is laid, unstable bricks are removed until an entire course proceeds unplaced, at which point the overall fitness is determined. If the fitness is greater than the previous generation, the genotype mutation is preserved. A lower fitness means that the mutation is discarded and the previous highest-fitness generation is mutated again. The cycle continues until the brick placement ratio reaches a specified amount; the net result is a stable, structurally sound brick tower (Figure 4).

RESULTS

Over the course of 205 generations, the tower depicted in Figure 6 grew from an initial height of 18 courses to a final, stable height of 120 courses, shown in Figure 7. Using an Intel
The first generation of an evolutionary sequence, initially using 418 bricks.

The 205th; over 8 minutes, the number of bricks has reached 3,517.

An early test produced while the algorithm was still undergoing calibration.

Core i7 processor running at 3.6 GHz, the structural optimization took roughly 8 minutes. More porous structures typically take longer; the evolution sequence depicted in Figure 4 took place over 2647 generations and roughly three and a half hours.

Although the solver is able to complete a tower with nearly every attempt, the net result is often somewhat under-built by physical standards—though the bricks are able to resist the vertical force enacted upon them by the solver, the algorithm does not yet simulate lateral loading such as wind. Since forces are applied on a per-object basis, this change will likely be relatively easy to implement.

Resisting this force may prove more difficult. The most obvious structural shortcoming of the solver’s current output is that the tower’s walls consist of only a single wythe of bricks. Adding two or more wythes may slow things down considerably; doubling the number of bricks significantly increases the computational overhead of each simulation.

A new technique for computational optimization may provide a way forward. Rather than simulate the entirety of the structure at all times, a set number of courses are passed to the physics engine; the remainder are stored as traditional Processing objects and are locked in space. At the time of writing, simulating the ten most recent courses seems to produce results indistinguishable from a fully simulated model, though this option will need to undergo sensitivity testing before being fully implemented. It is hoped that greater efficiencies will allow for further increases in realism and complexity.

**CONCLUSION**

Somewhat predictably, this evolutionary solver has the tendency to make towers more structurally conservative: porosity and formal variation are typically decreased over time. While the algorithm is correctly optimizing for structural stability, the results are relatively normative. Future explorations will focus on evolution towards more novel ends—how porous can a stable brick tower possibly be? An early trial may point the way forward. The tower depicted in Figure 8 was the unexpected outcome of an incorrect gravity value, but its radical porosity serves as a kind of working target for the next version of this solver.

**Postscript**

As it stands today, the cathedral at Beauvais is somewhat compromised—the vaulting, once ambitiously broad, is cluttered with a bird’s nest of ad-hoc structural steel reinforcement. Despite its aesthetic shortcomings, there is something poetic about this provisional arrangement; the intended cathedral was mutated and adapted over centuries, adopting the revisions of a host of designers. With its plurality of authors, one might say that Beauvais is more bazaar than cathedral (Raymond 1999). This is not a standalone case—architecture’s claim to authorship is inevitably muddled by the messy realities of construction. Any built project is an embodiment of concession: an authored work, sure, but one inflected (and often strengthened) by the unintended.

Ruin is perhaps the most extreme form of this concession. Though we must grapple with the demands of client, budget, or program, the architect’s most elemental struggle lies in resisting collapse. This paints materiality itself as a kind of stakeholder—bricks want something, after all, which is to fall to the ground. By incorporating the bricks themselves as secondary authors, it is hoped that this new approach helps to celebrate the gap between intent and execution. Maybe Kahn was right after all?
ACKNOWLEDGEMENTS
This project owes a great deal to the guidance of Benjamin Dillenburger, who provided valuable advice throughout its beginning stages.

REFERENCES


IMAGE CREDITS
All figures © Paul Harrison, 2016

Paul Harrison received his Master of Architecture from the University of Toronto, where he was awarded the Faculty Design Prize in 2015. He also holds a professional degree in mechanical engineering from Queen’s University; prior to his studies in architecture, he worked as a parametric design consultant for Rand Worldwide. Paul currently works as a project designer for Williamson Chong Architects.