FastForce: Real-Time Reinforcement of Laser-Cut Structures

MUHAMMAD ABDULLAH, MARTIN TARAZ, YANNIS KOMMANA, SHOHEI KATAKURA, ROBERT KOVACS, JOTARO SHIGEYAMA, THIJS ROUMEN, PATRICK BAUDISCH

Hasso Plattner Institute, University of Potsdam, Germany, firstname.lastname@hpi.de

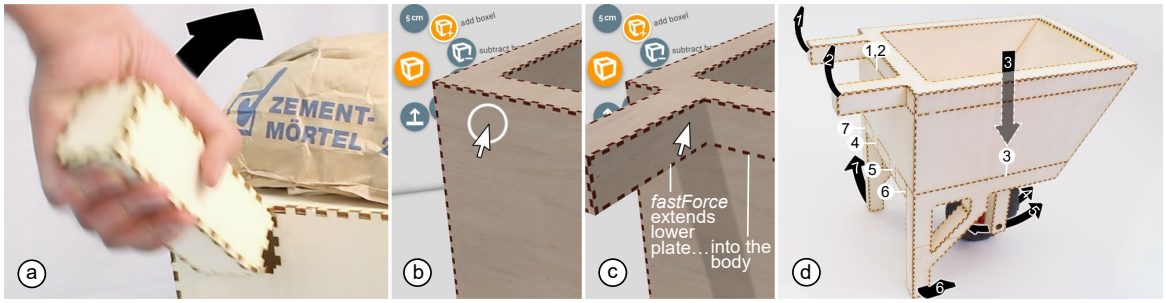

Figure 1: (a) The handle of this laser-cut wheelbarrow breaks, when the user is trying to lift a heavy load. (b) Our software tool fastForce addresses this. The moment the user clicks in the editor to create the handle, (c) not only the handle appears, but fastForce also detects the resulting point of failure and resolves it in real-time by extending the lower plate of the handle into the internal space of the wheelbarrow. (d) The resulting model has the same outer shape as the original model, yet allows lifting loads that are 45x times higher. Note that fastForce has detected additional points of failure (black arrows 2-7) and has generated corresponding reinforcement (white 2-7).

We present fastForce, a software tool that detects structural flaws in laser cut 3D models and fixes them by introducing additional plates into the model, thereby making models up to 52x stronger. By focusing on a specific type of structural issue, i.e., poorly connected sub-structures in closed box structures, fastForce achieves real-time performance (10^6x faster than finite element analysis, in the specific case of the wheelbarrow from Figure 1). This allows fastForce to fix structural issues continuously in the background, while users stay focused on editing their models and without ever becoming aware of any structural issues.

In our study, six of seven participants inadvertently introduced severe structural flaws into the guitar stands they designed. Similarly, we found 286 of 402 relevant models in the kyub [1] model library to contain such flaws. We integrated fastForce into a 3D editor for lasercutting (kyub) and found that even with high plate counts fastForce achieves real-time performance.

CCS CONCEPTS •Human-centered computing~Human computer interaction (HCI)~Interactive systems and tools

Additional Keywords and Phrases: Personal fabrication, lasercutting, structural analysis, structural reinforcement

ACM Reference Format:

Muhammad Abdullah, Martin Taraz, Yannis Kommana, Shohei Katakura, Robert Kovacs, Jotaro Shigeyama, Thijs Roumen and Patrick Baudisch. 2021. FastForce: Automatic reinforcement of laser-cut structures. In CHI Conference on Human Factors in Computing Systems (CHI '21), May 8–13, 2021, Yokohama, Japan. ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3411764.3445466

1 INTRODUCTION

Recently, researchers presented software systems that help users design 3D models for lasercutting, such as FlatFitFab [17], which helps users create interlocking cross sections. Kyub [1] builds on ideas from FlatFitFab and extends it towards structures that withstand 1000x larger loads. Kyub achieves this by producing "closed box structures" held together by box joints (finger joints if not perpendicular). As a result, kyub allows users to make chairs that people can sit on, even from comparably thin building materials.

Unfortunately, as the kyub authors acknowledge, in order to allow them to take such high loads, some models require "reinforcement", i.e., they require additional internal plates to be added to the model. While the kyub authors propose users add these reinforcement plates manually, our user study finds that users are unable to do so. In this study (see *qualitative study*), six of seven participants inadvertently introduced severe structural flaws into the guitar stands we had them design, none of them became aware of them, and so no flaws were fixed prior to fabrication. Similarly, we found 286 of 402 relevant user generated models in the kyub repository to contain such flaws (see *survey*). To illustrate this point, we encourage the reader to try to locate at least four easy ways of breaking the snowboarder sculpture shown in Figure 2a.

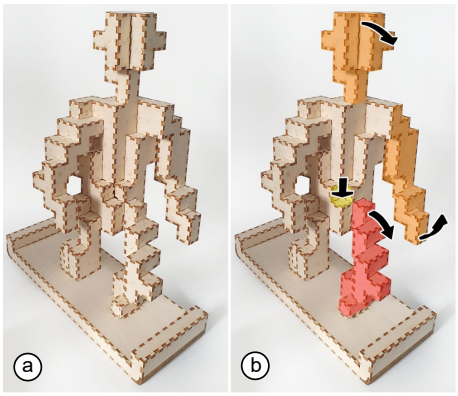

Figure 2: (a) Can you locate points of failure without looking at (b)? Here are four points of failure fastForce identifies for this model: pulling/pushing any of the red or orange parts in the direction suggested by the arrows will cause the respective part to fall inwards or rip off the rest of the model. FastForce identifies all four such points of failure, as well as another six only visible from the backside.

If users cannot figure out the need for reinforcement, reinforcement needs to be created automatically.

At first glance, such reinforcement functionality could be integrated into the user's workflow as a separate step *after* the design workflow. However, it is generally understood that such a design-first-then-engineer workflow leads to suboptimal results. For illustration consider Figure 3a where a designer is working on two competing designs of a guitar stand. (a) The designer likes this first design for its aesthetic qualities, but (b) keeps a backup design as fallback in case the preferred design is not sturdy enough. If the environment had real-time structural analysis (e.g. as seen in [22]), the user would see warnings *during* modelling. In this example, however, there are no warnings (and if we (c) flip the preferred guitar stand around we may see why, which is that it has already been reinforced automatically by our software). The absence of a warning allows the user to dismiss the fallback design early on and (d) start elaborating on the preferred design instead.

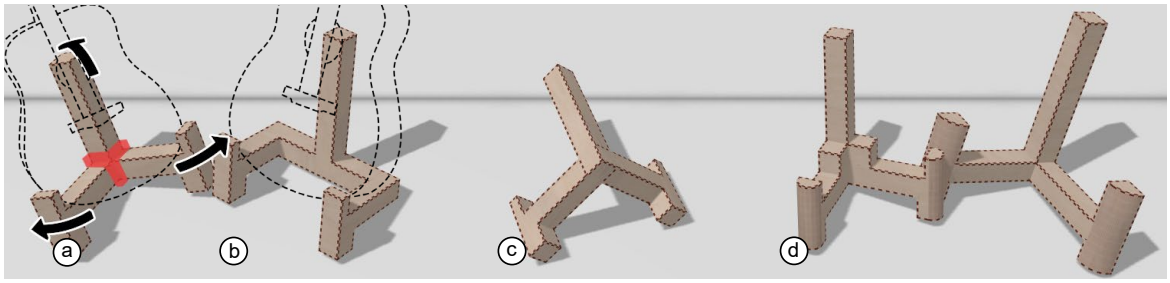

Figure 3: (a) The user's preferred guitar stand design competes with (b) their backup design. (c) The preferred model shows no warnings (and flipping it around shows why: it has already been reinforced automatically using our system *fastForce*). (d)This allows the user to drop the fallback design and focus the design exploration on the preferred design.

In this paper, we address these issues by generating reinforcement (1) automatically and (2) in real-time. While we started by exploring the traditional approach of finite element analysis [3], we found it to be too slow for interactive use. Thus we have developed a custom algorithm that finds and fixes points of failure in real-time by formulating them as a graph connectivity problem.

As shown in Figure 1b and c, our software tool fastForce runs in the background while users are editing their 3D model (here in kyub [1]). The moment the user starts modeling, in this case adding a handle for the wheelbarrow, fastForce simultaneously adds the necessary reinforcement, here by extending the handle's lower plate into the interior of the wheelbarrow. (d) Since fastForce only modifies the wheelbarrow's interior, the outer shape of the model is preserved while still allowing to lift a load that is 45.6 times higher. FastForce generates reinforcement for the other six points of failure similarly (black arrows 2-7).

2 CONTRIBUTIONS & LIMITATIONS

In this paper, we make five main contributions. First, we identify the fact that non-engineers are unable to identify structural issues in laser-cut closed box structures. We support this observation with a user study and an analysis of 402 models from an online repository (kyub). Second, we present a systematic classification of such issues. Third, we present an algorithm called fastForce that detects such issues automatically and reinforces them while users are working on their design. Fourth, we have integrated fastForce into an interactive 3D editor (Kyub [1]), where it runs in the background without interfering with users' modeling activity. Fifth, we conducted an runtime evaluation to support our claim of real-time performance and a technical evaluation that finds weakly connected sub-structures reinforced using fastForce withstand up to 52x larger forces.

FastForce is subject to three main limitations: First, rather than addressing arbitrary types of structural issues, fastForce addresses issues from laser-cut closed box structures breaking apart due poorly connected sub-structures. This makes sense though, as we find that these points of failure are critical. Second, reinforcement generated by fastForce is not always optimal. fastForce produces reinforcement to withstand arbitrary forces and therefore they may be less optimal than reinforcement designed to specifically support a force indicated by users. This was a conscious design decision we made, as it allowed us to eliminate the definition of loads from the user interface and thus reduce user interface complexity. Finally, fastForce does not cover the entire design space of today's 3D editors. In particular, fastForce only works for enclosed structures with unused volume.

3 FINITE ELEMENT ANALYSIS & UNDERSTANDING THE PROBLEM SPACE

We started our search for a reinforcement algorithm by turning to the canonical contender for structural analysis, i.e., finite element analysis (FEA) [3]. To explore whether FEA would solve our problem, we used the Ansys software suite [18] to simulate the effects of different forces on the wheelbarrow model. As shown in Figure 4, the analysis successfully reveals the points of failure in this model. For example the resulting visualization (which we here picked to show deformation rather than stress) shows, that the inside of the wheelbarrow falls in under load.

Unfortunately, however, the computation is far from being real-time (8 hours on a 16-core machine, 96 GB of RAM). In this particular case FEA is slow by about six orders of magnitude. While this factor would be smaller for simpler models that have fewer joints/surface area FEA would still not fulfill our requirement for interactive rates. The reason why FEA is slow is that it breaks down the 45 laser-cut plates into a larger number (here 96,330) of FEA nodes due to complex finger joint geometry. We used the default adaptive meshing provided by the Ansys mechanical software to generate the mesh for each plate; better computation time could be achieved by fine-tuning the mesh.

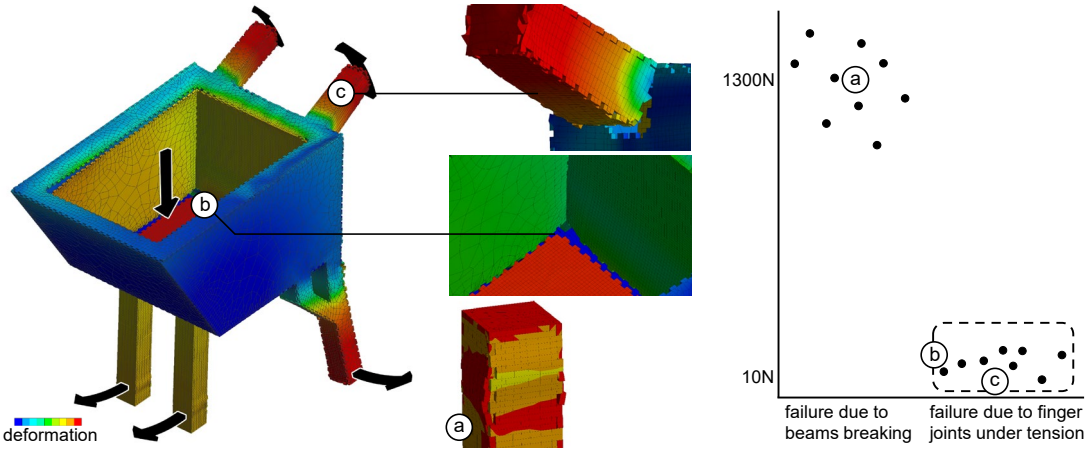

Figure 4: Results of the finite element analysis of the wheelbarrow from Figure 1 (visualization shows deformation, not stress). The failures can be divided in two classes, those caused by (a) entire beams breaking requiring considerable force, while others (b, c) caused by finger joints coming apart which break an order of magnitude earlier. Reinforcing the latter will substantially increase the force required to break the model.

With the objective of speeding up this computation, we classified the failures identified by finite element analysis. We realized that they fall into two distinct classes: (a) On one hand there were issues caused by entire beams breaking, which tends to happen when very high forces were being applied. On the other hand (b, c) we would see issues that caused the model to break under very small forces. Naturally, this second category is the one that required exploring: if we could solve only this category, this could elevate models to a point where the forces required to break them would be 10x-100x higher.

Closer inspection revealed that this category of "critical" failures originate when laser-cut closed box structures have deteriorated into poorly connected sub-structures. Figure 5 illustrates this. Consider the second case labeled "E_1"; it represents the wheelbarrow's handle. It breaks because only one of its plates is rigidly connected to the rest of the model; the other three plates, in contrast, are connected by means of finger joints only. When subjected to tension, the fingers are pulled out, resulting in the single remaining plate supporting the entire load. This causes it to break even under very small loads (here as small as ~9.8N).

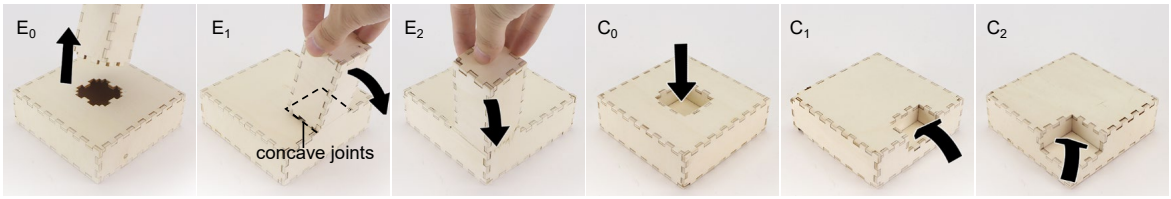

Figure 5: The six main types of structural issues of closed box structures.

Similar points of failure arise almost anywhere where two sub-structures of different sizes connect, or where sub-structures are offset with respect to each other and Figure 5 classifies the six main resulting cases, which we label by **E** for extrusions and **C** for cavities and then index by the number of remaining connecting plates.

An important point to note is that while all finger joints are weak against tension, not all of them can be subjected to tension without an external lever. An example of this would be a platonic cube, made from 6 plates connected with finger joints. They only way to apply tension to the joints would be to use an external level (e.g. a screw driver). However, as labelled in E_1, joints whose fingers meet on the inside of the model are *exposed* to tension. The reason is that they are always part of an extrusion or a cavity which provides a built in lever to apply tension at these finger joints. We will refer to these as concave finger joints.

As we present in detail later (section *validation*), while C_2 and E_2 have enough structural integrity left for some applications (the wheelbarrow's legs 6-7 in Figure 1d are of type E_2), C_1 and E_1, break quickly under even smaller loads (see Figure 20). However, C_0 and E_0 are the weakest points of failure and can break so easily that these types of objects tend to fail during regular handling.

As we report in more detail in our survey of the kyub model repository (see also Figure 17 in the *survey* section), all these cases occur in actual user generated models.

Based on this list of cases to search for, we propose an algorithm that would focus on detecting *only* these points of failure. This turned out to be the key to solving the real-time requirement, as it allowed us to reformulate the analysis as a graph connectivity problem making it computationally tractable.

We also used this reformulation as an opportunity to eliminate the notion of *loads* from our algorithm. The analysis of the wheelbarrow in Figure 4 required us to place seven loads (visualized as black arrows) reflecting not only that the wheelbarrow will be used to lift a load, but also that the front legs might get stuck and thus be subjected to bending moments. It quickly became clear that non-expert users would have a hard time placing loads—even more so if the model at hand was a toy in which case the actual forces expected during use would be largely unpredictable. By eliminating the notion of loads from the user interface we can integrate fastForce into a 3D editor in "headless" fashion, i.e., *without* adding to user interface complexity.

4 THE BUILDING BLOCKS OF REINFORCEMENT

We have manually devised reinforcement for the six points of failure cases, basing our principles on structural engineering, particularly on beam design [16].

As illustrated by Figure 6 at the example of the E_1 failure type, all types of reinforcements we designed create a reliable connection between the sub-structures using at least two parallel through plates. As shown, the reinforcement plates extend internally and hook into the sides while respecting the hull/envelope of the model. By hooking into the

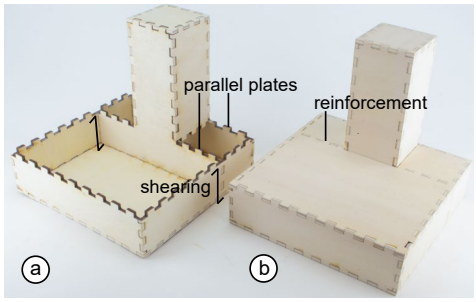

Figure 6: (a) A reliable connection is formed by at least two parallel plates extending internally and hooking into the sides. (b) The outer hull of the object remains the same.

sides the reinforcement plates allow finger joints to be loaded on compression and shearing (which they are strong against), counteracting tension force.

Figure 7 shows some of the reinforcement options we generated for the six individual failure cases. Note that there are always either two or three solutions for each case—this additional choice is essential as it allow fastForce to resolve assemblability issues (see *ensuring assemblability*).

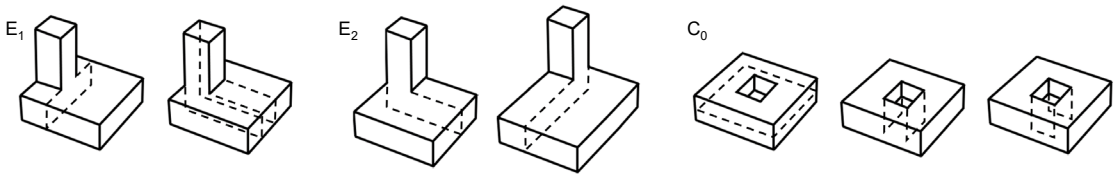

Figure 7: Some of the manually designed sets of reinforcement options.

The reason we focus on "two parallel plate connections" is that we get the maximum benefit out of two plate extensions as internal space is limited. Pairs of parallel plates can also reinforce convoluted models, such as the knot shown in Figure 8a.

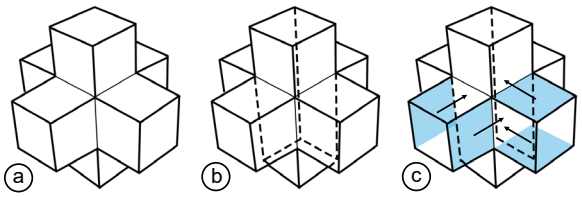

Figure 8: (a) Extending all four sides of one E_0 element (b) leaves no space to reinforce the other four. (c) The only way to reinforce all elements is to extend two plates per element.

5 THE FASTFORCE ALGORITHM

We now present the fastForce algorithm. FastForce detects all points of failure in the model, places appropriate reinforcement automatically and in real-time, and fine-tunes the reinforcement so as to guarantee assemblability.

FastForce achieves real-time performance by looking for weakly connected components in a graph representation of the model which point to weakly connected sub-structures. Since the graph structure is at a higher level of abstraction

than 3D geometry, a model that would become highly complex if represented in FEA continues to be represented by a small number of nodes and edges.

5.1 Step 1: FastForce identifies critical points of failure

As illustrated by Figure 9, fastForce starts by locating points of failure based on a simplified graph representation, here the wheelbarrow model from Figure 1. The undirected graph represents each 2D plate in the model as a node, while finger joints are represented as edges between nodes.

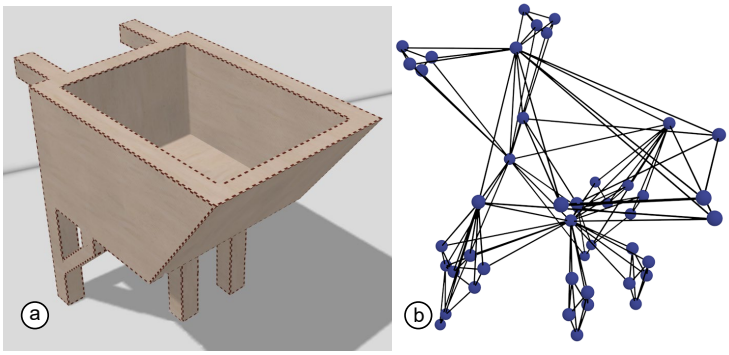

Figure 9: (a) FastForce represents this wheelbarrow as (b) an undirected graph. Nodes represent plates and edges represent joints.

Figure 10a the graph distinguishes between normal and concave finger joints (red edges), as they can be subjected to tension, causing potential failure. FastForce simulates their failure by *removing* all concave joints (i.e., all red edges) from the graph. As shown in Figure 10a and b this causes the graph to fall apart into four disconnected subgraphs. This corresponds to E_0 (or C_0) type points of failure i.e. these sub-structures were only connected by concave finger joints. This means that the corresponding physical model, the wheelbarrow, will also disconnect into four parts when subjected to minor forces.

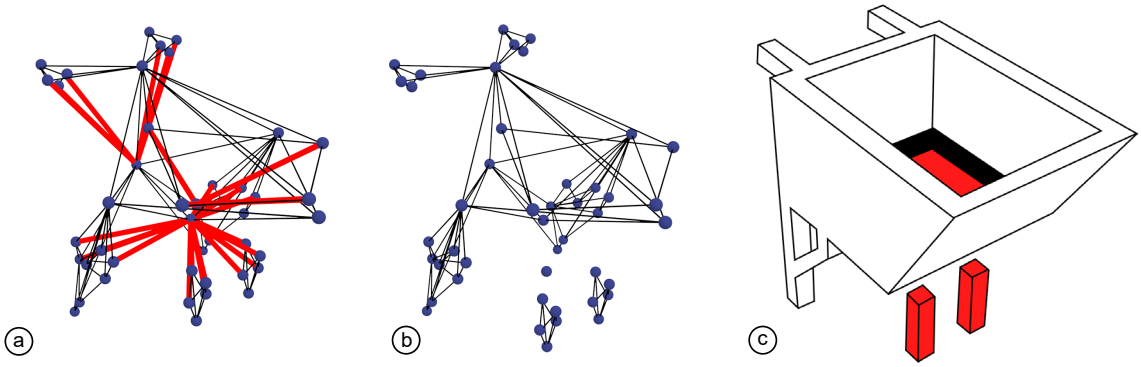

Figure 10: (a) When removing the red edges representing concave joints (b) this graph decomposes into four subgraphs, suggesting that (c) this model of a wheelbarrow will disconnect into four parts based on E_0 (or C_0) points of failure.

E_1 (or C_1) points of failure occur when sub-structures are only connected by a single plate. As illustrated by Figure 11, removing the red edges created three subgraphs connected by a single node. FastForce locates the problem by

performing a one-node cut on all the disconnected subgraphs. This results in a set of single nodes, which, if removed, will disconnect the subgraphs.

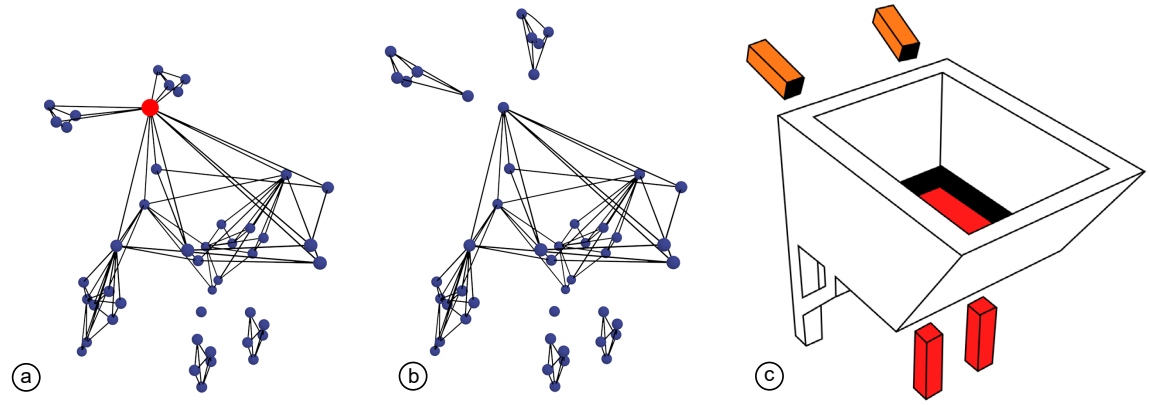

Figure 11: (a) When removing isolated nodes that connect subgraphs (b) this model graph decomposes into disconnected subgraphs, suggesting (c) that these parts are subject to E_1 (or C_1) points of failure.

Figure 12 shows that fastForce identifies E_2 (or C_2) points of failure by performing a two-node cut on the remaining subgraphs, meaning the back legs of the wheelbarrow are only held in place by two plates. Specifically, for E_2 (or C_2), fastForce checks if these two plates are connected to each other, forming the corner connection. If they are not connected and are parallel, they fulfill our notion of a reliable connection and will not benefit from further reinforcement.

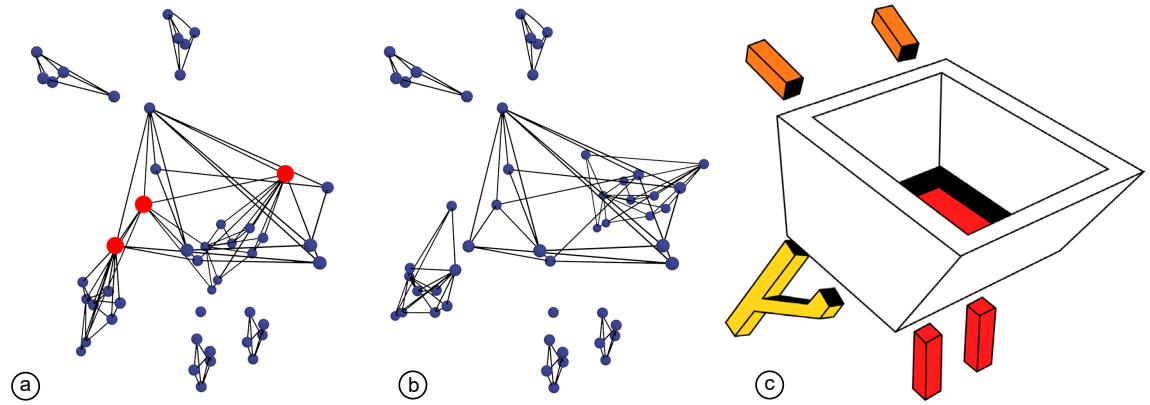

Figure 12: (a) When removing pairs of nodes that connect subgraphs (b) this model graph decomposes further, suggesting (c) that these parts are subject to E_2 (or C_2) points of potential failure.

We summarize this step in algorithm 1. The algorithm has a linear complexity of $O(n + e)$, where n are the number of nodes (plates) and e are the number of edges (joints). Node-cuts are based on the *Hopcroft-Tarjan* Algorithm [8].

5.2 Step 2: FastForce adds reinforcement

FastForce adds reinforcement sequentially starting with the weakest points of failure E_0 (and C_0). After selecting a point of failure, fastForce identifies the set of plates that can be extended for reinforcement. FastForce achieves this by

revisiting the removed graph edges (concave joints) that originally belonged to the disconnected subgraph. These edges connect to nodes, which point to their respective plates represented as polygons at the mesh level. These plates connect the two sub-structures together and are extended to reinforce them.

ALGORITHM 1: locating points of potential failure

```
Input: Connected Plate Graph g
Output: ToReinforce[ ]
ToReinforce ← new Array ( )
for each edge ∈ g do
  if isConcave (edge) then
    g.removeEdge(edge)
  end
end
c=g.getConnectedComponents ( )
for each (component1, component2) ∈ c × c, component1≠ component2 do
  j=findConnectingJoints(g, component1, component2)
  if notEmpty (j) then
    toReinforce.push(j)
  end
end
for each component ∈ c do
  w'=findWeaklyConnectedComponents(comp, 1)
  for each (w1, w2) ∈ w', w1 ≠ w2 do
    j=findConnectingJoints(g, w1, w2)
    if notEmpty (j) then
      toReinforce.push(j)
    end
  end
  for each comp' ∈ w' do
    w''=findWeaklyConnectedComponents(comp', 2)
    for each (w1, w2) ∈ w'', w1 ≠ w2 do
      j=findConnectingJoints(g, w1, w2)
      if notEmpty (j) then
        toReinforce.push(j)
      end
    end
  end
end
```

Based on the point of failure fastForce chooses which parallel plate to extend (options are shown in Figure 7). FastForce gives preference to reinforce using the minimum number of plate extensions. If all options result in the same number of plates being extended, fastForce chooses the longer reinforcement to maximize leverage.

Figure 13 shows the process of reinforcing the "tray" of the wheelbarrow. (a) After choosing a plate to extend, a slice plane is generated which intersects with the surrounding geometry. (b) This forms a closed intersection path which is used to create an extension to the original plate. This new reinforcement plate is added to the mesh. (c) FastForce creates joints, where the extended plate intersects with other plates. (d) FastForce then repeats this process for each point of failure identified in the first part of the algorithm.

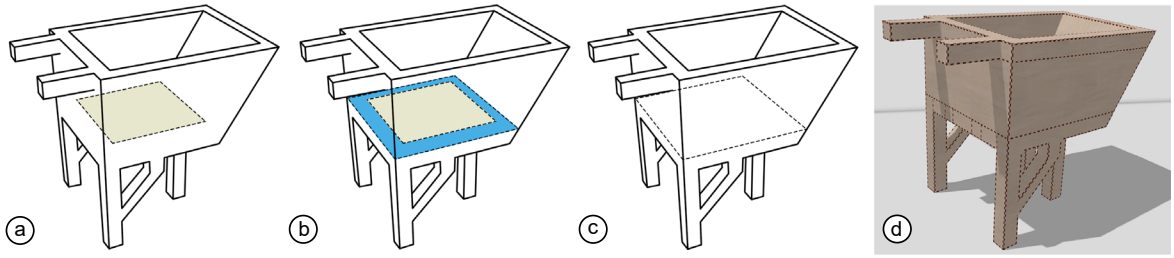

Figure 13: (a) FastForce reinforces the "tray" by (b) extending the chosen plate using a slice plane. (c) The plate is extended and joints are created where it intersects the other plates. (d) FastForce, sequentially reinforces the remaining points of failure.

After the tray, the "front braces" are reinforced by extending and anchoring two of their parallel side plates into the body of the wheelbarrow. They hook into the far sides ensuring they cannot be pulled out easily. The handles are reinforced by extending their bottom plates and (d) the "rear braces" are reinforced by extending their side plates.

5.3 Ensuring Assemblability

In step 2 when choosing a plate to extend fastForce always make sure that the resulting reinforcement plates do not generate an interlocked closed loop, as this can lead to problems during assembly. For example the "wall mount" model shown in Figure 14a cannot be physically assembled as two plates create an interlocked loop. FastForce resolves this by replacing the reinforcement with the other orientation. For the majority of models, e.g. Figure 14b, this solves the issue.

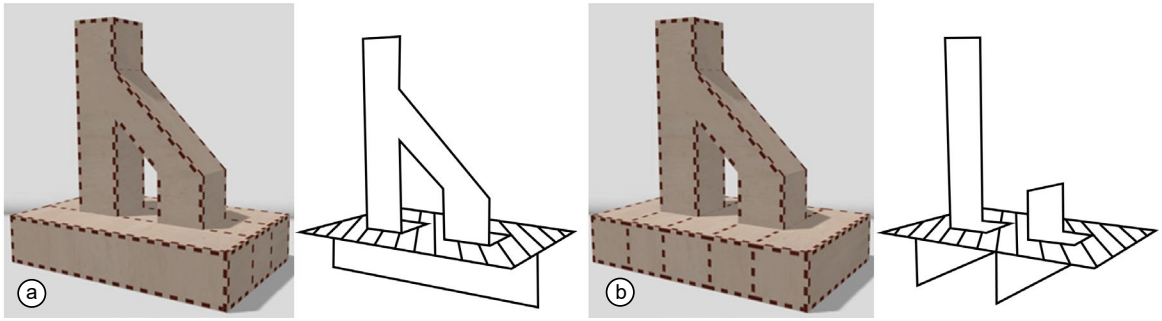

Figure 14: In case of this wall mount, (a) if the extended plates form closed loops, the physical object can no longer be assembled. FastForce detects this and (b) reinforces in the other direction to solve this problem.

Figure 15 shows an example where changing the orientations (a and b) lead to the same issue. FastForce detects such cases and responds by (c) dropping reinforcement elements in (outside-in order). Note that the resulting structure despite

the reduced reinforcement, is still structurally sound. The reason is that while the local elements have a single plate connecting them (E_1), the complete substructure is still supported by two parallel plates.

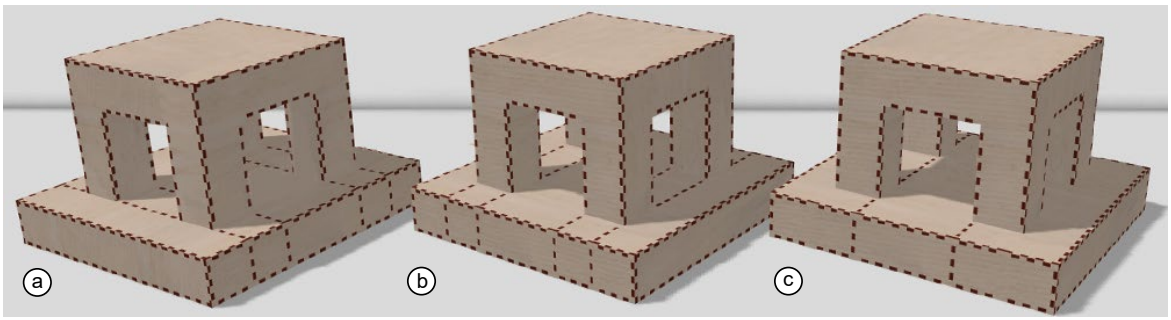

Figure 15: (a) This model cannot be assembled, even when (b) the reinforcement is rotated. (c) *FastForce* responds by reducing the reinforcement, which still fulfills our notion of a reliable connection using two parallel plates.

6 INTERACTION MODEL AND INTEGRATION INTO 3D EDITOR

We integrated fastForce into a 3D editor for lasercutting (kyub [1]), allowing fastForce to be used interactively. Figure 16 illustrates this with an example of a user designing a wall mount for a bench. (a) After *stretching* the kyub's default boxel into a bigger box, (b) the user adds an extrusion using the *add boxel* tool, but rather than showing the extrusion in isolation, (c) the reinforcement appears instantaneously with the extrusion. (d) As the user adds another extrusion, the system adds reinforcement for that as well, and combines them together. (e) When the user connects the two extrusions, (f) FastForce responds instantaneously by re-orienting the reinforcement to prevent the interlocking loop, allowing the model to be (g) laser-cut and assembled.

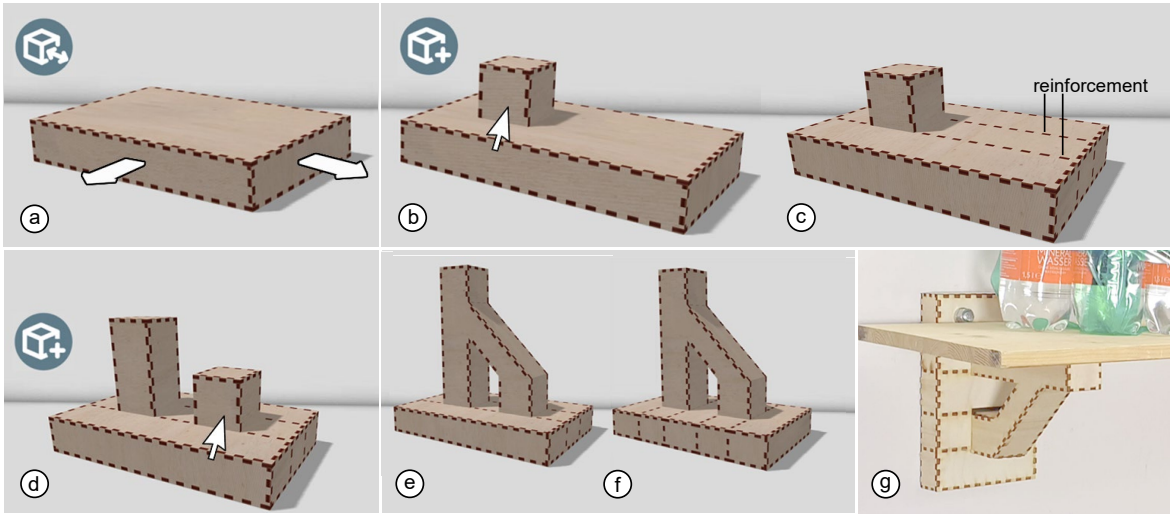

Figure 16: User session showing the creation of a wall mount using Kyub with fastForce integration.

The key contribution we want to showcase in this walkthrough, is that there is *no* fastForce-specific user interface. Reinforcement shows up without any specific user interaction. Users can *tell* that this model has been reinforced, based

on seeing the additional finger joints. These finger joints also serve as a user interface for removing the reinforcement, i.e., users may click the additional finger joints with kyub's minus tool to delete the reinforcement in the rare case that users need control over the inside of the model (boxes, containers, or molds). This turns off the automatic reinforcement so that it does not try to insert the plate again.

7 SURVEY: USER CREATED CONTENT IS SUBJECT TO POINTS OF FAILURE

We collected user generated models from an online repository hosted by kyub, some of which are shown in Figure 17. The color corresponds to the type and severity of weakness with red being the most severe.

We started out with 1067 models. Eliminating structurally trivial models (prisms), left us with 402 models that we analyzed. Out of these 402 models, 286 models (71%) were subject to one or more critical point of failure and would benefit from fastForce's reinforcement.

Out of the remaining 116 models, 88 already had a reliable connection with more than two plates connecting the sub-structures. While 28 models either required living hinges to be extended for reinforcement or cutouts/decorative elements were present in the locations where the reinforcements would hook in.

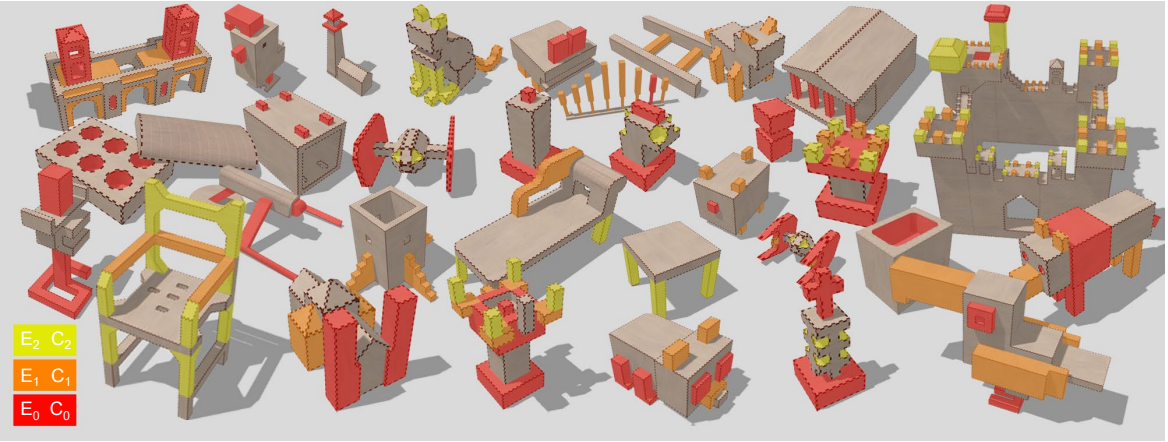

Figure 17: A selection of user generated models (kyub.com) that would benefit from reinforcements generated by fastForce. Red indicates E_0/C_0 type of weakness, meaning the highlighted part will disconnect completely when subjected to minor force. Orange and yellow indicate E_1/C_1 and E_2/C_2 points of failure respectively.

8 VALIDATION

8.1 Performance

We benchmarked the runtime performance of the kyub 3D editor after integrating fastForce. The editor was running on a lightweight laptop computer (X1 Carbon, 4-core machine, 10 GB of ram). Eight different models representing different plate counts were used. We simulated user interaction by performing a basic kyub command (push/pull) to modify the model and measured the response time. The results are summarized in Table 1.

The "Snowboarder" model was close to the worst-case scenario as only 5 out of 1067 models we tested had plate counts higher than 100. Thus, typical performance will be better than 47ms. The results show that even for models with high plate counts integrating fastForce still allows good response times, allowing the user to smoothly use the editor.

Table 1: Performance

Model	Number of Plates	Response time (ms)
Wheelbarrow (Figure 1)	30	10
Snowboarder (Figure 2)	196	118
Chair (Figure 17: bottom line)	46	21
Cathedral	52	11
Chess Queen	68	68
Bear	72	34
Chess King	86	44
Toucan	95	47

8.2 Qualitative study: users are unaware of points of failure

We used a workshop event on lasercutting to conduct a simple qualitative study to test our hypothesis that users are unaware of points of failure in the closed box structures they design.

As part of the workshop, we asked the seven non-engineer participants to design a guitar stand in kyub. Participants were told to design their guitar stands with the intent to fabricate and use them. The average age of the participants (three female) was 30 years. Six participants were teachers at a high school, while one participant was a bachelor student. None of the participants had any prior experience with laser cutters or modelling objects for lasercutting.

As training, participants were shown a simple reference guitar stand and participants replicated the design in kyub, which all participants completed in under 10 minutes. Participants were then given 30 minutes to design their own guitar stands.

Results: All participants succeeded at designing a guitar stand. Figure 18, shows the resulting designs. We now used fastForce to analyze these designs and found that six out of the seven designs had at least one point of failure that could easily break the object. Two designs (a) and (e) had problems where cavities would allow plates to be pushed in (C_0), while four designs (b-d, f) would potentially break if a bending moment was applied (E_1 and E_2). Only design (g), largely due to its simplistic nature in the form of an extruded prism, did not suffer from problems.

None of the participants were aware of these structural flaws in their design during modelling.

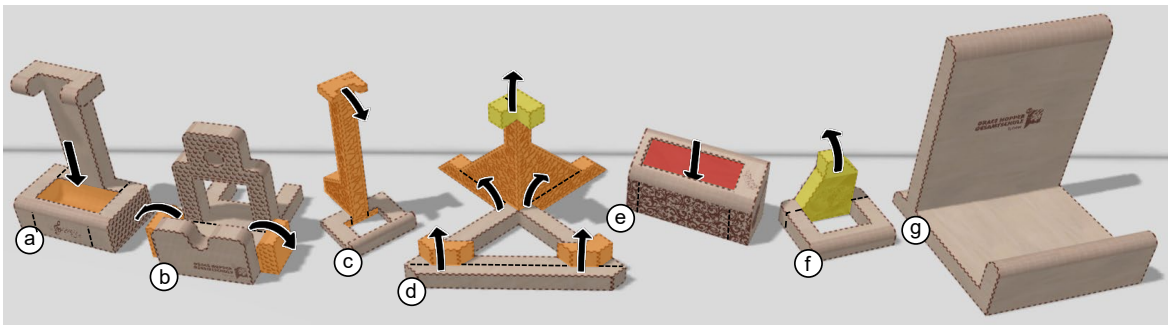

Figure 18: The guitar stands that were made by participants during the user study. Orange and yellow highlights indicate points of potential failure that users were unaware of during modelling. Black lines indicate reinforcements that fastForce places.

8.3 Testing the structural strength of models reinforced using fastForce

To evaluate the effect reinforcement created by fastForce, we tested the structural strength of 8 models (shown in Figure 20) after reinforcing them using fastForce.

We manufactured all models from standard 4mm 3-layer poplar plywood. We then used the custom rig with force sensor *forceX 2.30* made from aluminum profiles shown in Figure 19. In case of the extrusion models, we applied a bending moment from the side to the top of the extrusion. For models with a cavity, we applied a force from above the cavity to push them inwards. In both cases this results in tension forces acting at the finger joints susceptible to tension. For the wheelbarrow and the wall mount, we applied force in the direction of their intended use case.

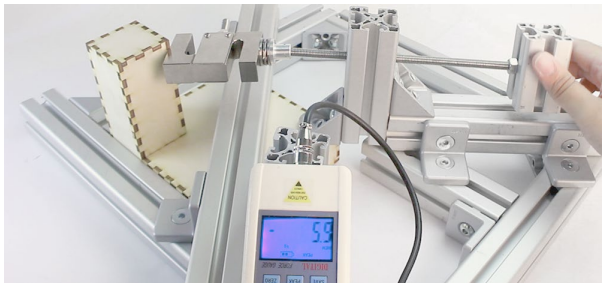

Figure 19: The test apparatus.

Results: Figure 20 shows the results. While reinforcement made the reasonably connected sub-structures E_2 and C_2 even stronger by 3x, as expected reinforcement using fastForce showed its biggest effect on the C_0 and E_0 designs, where they led to *improvements of 50x and 52x* respectively. These results confirm the fact that reinforcing laser-cut closed box structures using parallel plates considerably increases their structural strength.

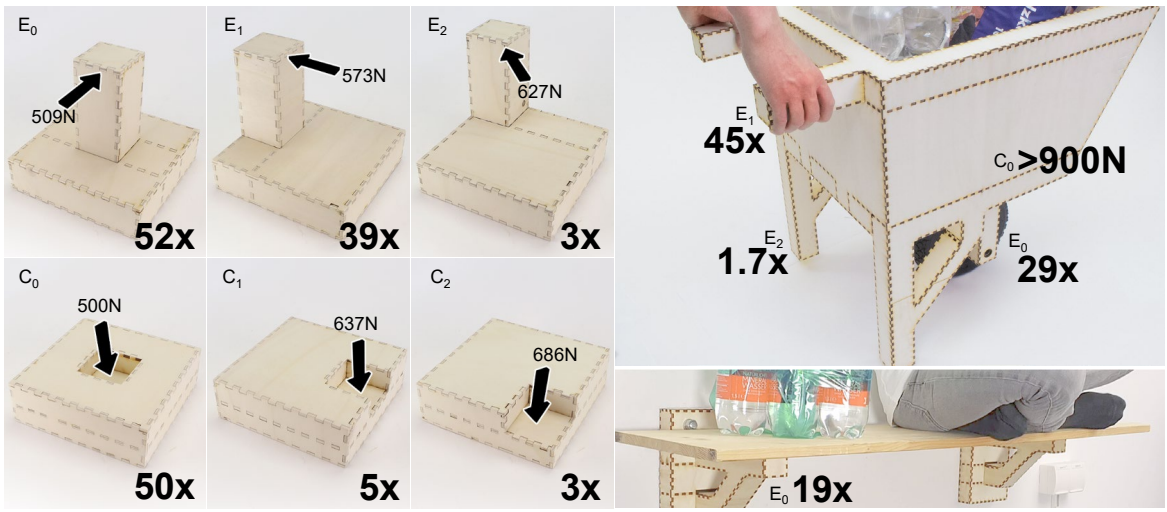

Figure 20: The reinforced test objects. Numbers indicate the factor of increase in strength after fastForce's reinforcement.

9 RELATED WORK

Our work builds on research into structural analysis, (dis)assembly in fabrication, systems for lasercutting, and personal fabrication based on closed box structures.

9.1 Finding structural weaknesses

Finite element analysis (FEA) [3] is traditionally used to validate the structural strength in various forms of construction. It has been used extensively to find structural problems for objects made in personal fabrication. Stress-relief [27] is a system that uses FEA to automatically fix structural problems in 3D models. Breuß et al. [4] use a skeletonization approach to enhance the stability of 3D printed structures and validate their results using FEM. Similarly, Lu et al. [13] built a system that optimizes the strength to weight ratio, they use FEA to validate assembled objects.

When building large scale structures, finding the structural weaknesses is of crucial importance for the structure to hold up. TrussFab [10] uses 3D printers to make such large structures and thus heavily relies on finite elements to verify the structure. Set-In-Stone [24] uses an extended FEM approach to perform structural optimization of large-scale binder-jetted structures. Chopper [14] uses some FEA but attempts to avoid it where possible because of its time-consuming nature. Schulz et al. [23] elegantly handle this limitation by running the FEA in pre-processing and then letting users quickly vary design parameters in CAD models.

Some scholars have utilized other techniques to circumvent the non-interactive rates of finite elements. Specifically for trusses, Markis et al. [15] present a real-time algorithm that identifies structural weaknesses. SketchChair [22] for example achieves interactive rates by employing a ragdoll physics model running in the background. Umetani and Schmidt [29] leverage the Euler Bernoulli method to find critical weaknesses in models. Langlois et al. [11] developed a stochastic method that enables real-time structural analysis for context aware fabrication. Umetani et al. [28] propose using real-time physics simulations to enable interactive furniture design. Shape structuralizer [5] uses sub-modeling to achieve interactive rates allowing users to create structurally sound objects through an iterative dialogue.

9.2 (Dis)assembly in fabrication

Fu et al. [7] developed interlocking furniture. To do this, they developed an algorithm that checks whether an object can be disassembled by taking parts out of the assembly piece by piece. Fabrication-aware Design with Intersecting Planar Pieces [25] leverages that method as well, to check which parts are free to move allowing a possible way to assemble the object. Both approaches are similar in their logic (finding out how to gradually disassemble a model), which inspired our general method of finding points of potential failure as well.

Graph analysis is also used by Desia [30] to find interlocking assemblies and Yao et al. [31] use directional blocking graphs to find possible assembly sequences.

9.3 Systems for lasercutting

2D SVGs are the typical format for sharing lasercutting plans and as such many lasercutting system such as Constructable [19] and Joinery [32] allow users to design and interact in 2D.

Since it can be hard for users to design 3D objects in 2D, many systems like CofiFab [26] and CutCAD [9] provide different forms of 3D modelling tools for lasercutting. FlatFitFab [17] allows users to model in 3D and implements their designs as 2D plates connected by cross joints. Similarly, Kyub [1] allows users to model in 3D using box based primitives and exports their models as 2D plates connected by finger joints. Platener [2] converts models made for 3D printing for lasercutting. Systems such as enclosed [31] and make-a-box [33] specialize in creating enclosures while [6] provides a system for designing mechanical characters.

9.4 Fabrication based on closed box structures

Platform framing, that uses wooden frames connected together into box-like structures, has been used for a while as a sturdy construction method for building homes [12]. It has also been paired with digital fabrication to facilitate construction in rural areas [20]. Closed box structures are also commonly used in the furniture industry, e.g. IKEA. This type of furniture is typically made from particle boards and reinforced using an internal honeycomb structure.

In Kyub [1], Baudisch et al. used the underlying strength of closed box structures to make sturdy laser-cut objects. The system uses plates connected together using finger joints to create self-supporting facades that can resists loads close to 4900N. Box like structures are also being introduced in 3D printing. In Boxception [21], Sajadi et al. propose 3D printing objects using box like cells to create impact resistance objects.

10 CONCLUSION AND FUTURE WORK

In this paper, we presented fastForce, a software tool designed to identify and remove structural weaknesses in laser-cut objects based on closed box structures. We identified the fact that non-engineers are unable to identify points of failure in laser-cut closed box structures. We presented a systematic classification of points of failure, and presented an algorithm called *fastForce* that detects them automatically and reinforces them *while* users are working on their design. We integrated fastForce into the interactive 3D editor kyub, where fastForce's real-time ability allows it to run in the background without interfering with users' modeling activity. We evaluated the run-time behavior of fastForce and found that even for models with high plate counts it is real-time capable. Finally, we evaluated the reinforcements added by fastForce and found that reinforced closed box structures can withstand up to 52x larger forces.

As future work, we plan on extending fastForce's capability of real-time reinforcement to non-closed box structures as well.

ACKNOWLEDGEMENTS

We would like to thank Dr. Michael Drass for his help with Ansys mechanical.

REFERENCES

[1] Patrick Baudisch, Arthur Silber, Yannis Kommana, Milan Gruner, Ludwig Wall, Kevin Reuss, Lukas Heilman, Robert Kovacs, Daniel Rechlitz, and Thijs Roumen. 2019. Kyub: A 3D Editor for Modeling Sturdy Laser-Cut Objects. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19). ACM, New York, NY, USA, Paper 566, 12 pages. DOI: https://doi.org/10.1145/3290605.3300796

[2] Dustin Beyer, Serafima Gurevich, Stefanie Mueller, Hsiang-Ting Chen, and Patrick Baudisch. 2015. Platener: Low-Fidelity Fabrication of 3D Objects by Substituting 3D Print with Laser-Cut Plates. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15). Association for Computing Machinery, New York, NY,USA, 1799–1806. DOI:https://doi.org/10.1145/2702123.2702225

[3] René de Borst, Mike A. Crisfield, Joris JC Remmers, and Clemens V. Verhoosel. *Nonlinear finite element analysis of solids and structures*. John Wiley & Sons, 2012.

[4] Michael Breuß, Johannes Buhl, Ashkan Mansouri Yarahmadi, Markus Bambach, and Pascal Peter. "A Simple Approach to Stiffness Enhancement of a Printable Shape by Hamilton-Jacobi Skeletonization." Procedia Manufacturing 47 (2020): 1190-1196.

[5] Subramanian Chidambaram, Yunbo Zhang, Venkatraghavan Sundararajan, Niklas Elmqvist, and Karthik Ramani. 2019. Shape Structuralizer: Design, Fabrication, and User-driven Iterative Refinement of 3D Mesh Models. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19). ACM, New York, NY, USA, Paper 663, 12 pages. DOI: https://doi.org/10.1145/3290605.3300893

[6] Stelian Coros, Bernhard Thomaszewski, Gioacchino Noris, Shinjiro Sueda, Moira Forberg, Robert W. Sumner, Wojciech Matusik, and Bernd Bickel. 2013. Computational design of mechanical characters. ACM Trans. Graph. 32, 4, Article 83 (July 2013), 12 pages. DOI:https://doi.org/10.1145/2461912.2461953

[7] Chi-Wing Fu, Peng Song, Xiaoqi Yan, Lee Wei Yang, Pradeep Kumar Jayaraman, and Daniel Cohen-Or. 2015. Computational interlocking furniture assembly. ACM Trans. Graph. 34, 4, Article 91 (July 2015), 11 pages. DOI: https://doi.org/10.1145/2766892

[8] Carsten Gutwenger, and Petra Mutzel. "A linear time implementation of SPQR-trees." In International Symposium on Graph Drawing, 2000.

[9] Florian Heller, Jan Thar, Dennis Lewandowski, Mirko Hartmann, Pierre Schoonbrood, Sophy Stönner, Simon Voelker, and Jan Borchers. 2018. CutCAD - An Open-source Tool to Design 3D Objects in 2D. In Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18).

Association for Computing Machinery, New York, NY, USA, 1135–1139. DOI:https://doi.org/10.1145/3196709.3196800

[10] Robert Kovacs, Anna Seufert, Ludwig Wall, Hsiang-Ting Chen, Florian Meinel, Willi Müller, Sijing You, Maximilian Brehm, Jonathan Striebel, Yannis Kommana, Alexander Popiak, Thomas Bläsius, and Patrick Baudisch. 2017. TrussFab: Fabricating Sturdy Large-Scale Structures on Desktop 3D Printers. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17). ACM, New York, NY, USA, 2606-2616. DOI: https://doi.org/10.1145/3025453.3026016

[11] Timothy Langlois, Ariel Shamir, Daniel Dror, Wojciech Matusik, and David I. W. Levin. 2016. Stochastic structural analysis for context-aware design and fabrication. ACM Trans. Graph. 35, 6, Article 226 (November 2016), 13 pages. DOI: https://doi.org/10.1145/2980179.2982436

[12] Li-Chu Lin. "The adaptability of two-by-four wood framing construction." In International Conference On Adaptable Building Structures. 2006.

[13] Lin Lu, Andrei Sharf, Haisen Zhao, Yuan Wei, Qingnan Fan, Xuelin Chen, Yann Savoye, Changhe Tu, Daniel Cohen-Or, and Baoquan Chen. 2014. Build-to-last: strength to weight 3D printed objects. ACM Trans. Graph. 33, 4, Article 97 (July 2014), 10 pages. DOI: https://doi.org/10.1145/2601097.2601168

[14] Linjie Luo, Ilya Baran, Szymon Rusinkiewicz, and Wojciech Matusik. 2012. Chopper: partitioning models into 3D-printable parts. ACM Trans. Graph.31, 6, Article 129 (November 2012), 9 pages. DOI: https://doi.org/10.1145/2366145.2366148

[15] Michael Makris, David Gerber, Anders Carlson, and Doug Noble. "Informing design through parametric integrated structural simulation: iterative structural feedback for design decision support of complex trusses." (2013).

[16] Dan B. Marghitu, *Mechanical engineer's handbook*. Elsevier, Chapter 3: Stress, 120–147, 2001.

[17] James McCrae, Nobuyuki Umetani, and Karan Singh. 2014. FlatFitFab: interactive modeling with planar sections. In Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST '14). Association for Computing Machinery, New York, NY, USA, 13–22. DOI:https://doi.org/10.1145/2642918.2647388

[18] Ansys Mechanical, https://www.ansys.com/products/structures, last accessed September 2020.

[19] Stefanie Mueller, Pedro Lopes, and Patrick Baudisch. 2012. Interactive construction: interactive fabrication of functional mechanical devices. In Proceedings of the 25th annual ACM symposium on User interface software and technology (UIST '12). Association for Computing Machinery, New York, NY, USA,599–606. DOI:https://doi.org/10.1145/2380116.2380191

[20] Lawrence Sass, and Marcel Botha. "The instant house: a model of design production with digital fabrication." *International Journal of Architectural Computing* 4, no. 4 (2006): 109-123.

[21] Seyed Mohammad Sajadi, Peter Samora Owuor, Robert Vajtai, Jun Lou, Ravi Sastri Ayyagari, Chandra Sekhar Tiwary, and Pulickel M. Ajayan. "Boxception: Impact Resistance Structure Using 3D Printing." *Advanced Engineering Materials*(2019): 1900167.

[22] Greg Saul, Manfred Lau, Jun Mitani, and Takeo Igarashi. 2010. SketchChair: an all-in-one chair design system for end users. In Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction (TEI '11). ACM, New York, NY, USA, 73-80. DOI=http://dx.doi.org/10.1145/1935701.1935717

[23] Adriana Schulz, Jie Xu, Bo Zhu, Changxi Zheng, Eitan Grinspun, and Wojciech Matusik. 2017. Interactive design space exploration and optimization for CAD models. ACM Trans. Graph. 36, 4, Article 157 (July 2017), 14 pages. DOI: https://doi.org/10.1145/3072959.3073688

[24] Christian Schumacher, Jonas Zehnder, and Moritz Bächer. 2018. Set-in-stone: worst-case optimization of structures weak in tension. ACM Trans. Graph. 37, 6, Article 252 (December 2018), 13 pages. DOI:https://doi.org/10.1145/3272127.3275085

[25] Yuliy Schwartzburg, and Mark Pauly. "Fabrication-aware design with intersecting planar pieces." In *Computer Graphics Forum*, vol. 32, no. 2pt3, pp. 317-326. Oxford, UK: Blackwell Publishing Ltd, 2013.

[26] Peng Song, Bailin Deng, Ziqi Wang, Zhichao Dong, Wei Li, Chi-Wing Fu, and Ligang Liu. 2016. CofiFab: coarse-to-fine fabrication of large 3D objects. ACM Trans. Graph. 35, 4, Article 45 (July 2016), 11 pages. DOI:https://doi.org/10.1145/2897824.2925876

[27] Ondrej Stava, Juraj Vanek, Bedrich Benes, Nathan Carr, and Radomír Měch. 2012. Stress relief: improving structural strength of 3D printable objects. ACM Trans. Graph. 31, 4, Article 48 (July 2012), 11 pages. DOI: https://doi.org/10.1145/2185520.2185544

[28] Nobuyuki Umetani, Takeo Igarashi, and Niloy J. Mitra. 2012. Guided exploration of physically valid shapes for furniture design. ACM Trans. Graph.31, 4, Article 86 (July 2012), 11 pages. DOI: https://doi.org/10.1145/2185520.2185582

[29] Nobuyuki Umetani and Ryan Schmidt. 2013. Cross-sectional structural analysis for 3D printing optimization. In SIGGRAPH Asia 2013 Technical Briefs(SA '13). ACM, New York, NY, USA, Article 5, 4 pages. DOI: https://doi.org/10.1145/2542355.2542361

[30] Ziqi Wang, Peng Song, and Mark Pauly. 2018. DESIA: a general framework for designing interlocking assemblies. ACM Trans. Graph. 37, 6, Article 191 (December 2018), 14 pages. DOI:https://doi.org/10.1145/3272127.3275034

[31] Christian Weichel, Manfred Lau, and Hans Gellersen. 2013. Enclosed: a component-centric interface for designing prototype enclosures. In Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction (TEI '13). Association for Computing Machinery, NewYork, NY, USA,215 218. DOI:https://doi.org/10.1145/2460625.2460659

[32] Clement Zheng, Ellen Yi-Luen Do, and Jim Budd. 2017. Joinery: Parametric Joint Generation for Laser Cut Assemblies. In Proceedings of the 2017 ACM SIGCHI Conference on Creativity and Cognition (C&C '17). Association for Computing Machinery, New York, NY, USA, 63–74. DOI:https://doi.org/10.1145/3059454.3059459

[33] Make a box, https://makeabox.io, last accessed September 2020.