Participatory Layout Design-Fabrication with Native Forms of Natural Materials

Figure 1: *Left: an overview of the workflow: 1. fix branches on plates. 2. scan the plates and upload the model. 3. play the game with scanned branches. 4. fabricate joineries by a CNC router. Right top: branch layouts partially designed by participants in a workshop. Right bottom: the fabricated 2D fence (2m x 0.9m). Each pair of branches is connected with rigid lapped joinery.*

Abstract

Diverse natural materials such as stones and woods have been used as architectural elements, preserving their native forms since primitive shelters, however, the use of them in modern buildings is limited due to their irregular properties. In this paper, we take the diversity as playful inputs for design task, and present our game-based design-fabrication platform for customized architectural elements. Taking tree branches as a material with native forms, our online game *BranchConnect* enables users to design 2D networks of branches and fabricate it by a CNC router. The game considers fabrication constraints such as limitations with ordinal 3-axis CNC routers. Each connection has a customized unique joinery adapted to native forms of branches. The scoring system of the game guides users to design structurally sound solutions with given branches. Together with low-cost mobile scanning devices, users with diverse contexts can contribute to design and fabrication process not only as a game user, but also from collecting branches around their physical environments and uploading them to our online platform. For validating our process, we conducted a workshop with end-users (children and their parents). They collected branches in a nearby forest and contributed to design/fabricate a 2D fence with our system.

Keywords: radiosity, global illumination, constant time

Concepts: •**Computing methodologies** → **Image manipulation;** *Computational photography;*

SIGGRAPH 2016 Posters, July 24-28, 2016, Anaheim, CA
ISBN: 978-1-4503-ABCD-E/16/07
DOI: http://doi.acm.org/10.1145/9999997.9999999

1 Introduction

Modern buildings are characterized by its uniformity; built upon the same principle of construction system, consisting of standardized building component and its assembly process. The standardized construction system is favored because of its efficiency in design and production. Each component satisfies specified structural performance, thus an assembled structure can be analyzed systematically. On the other hand, excessive standardization converged buildings into similar materials and details, resulting in the detached design from built environment. Reacting on the issue, designers and architects actively use local materials not only as inspirational sources, but also as a catalyst of their design to the local context [Oliver 1997].

Since primitive shelters and huts, traditional construction has used locally found natural materials directly as they are found [Weston 2003]. Such a direct use can not compete with highly standardized materials and construction system, however, the uniqueness of native forms is a valuable quality which is lacking in standardized materials. As the material is locally obtained, building and living get much closer, thus people using the building can easily commit design and fabrication, fostering the sense of belonging to the community []. TODO: citation Locally obtained materials can easily connect design and the context of built environment in this way, but their irregular properties limit the use of them in modern buildings. Traditionally, craftsman has taken care irregular natural materials varying their native forms and dynamically design the global design by considering individual material properties [Pye 1968]. Such a task is difficult to be automated and these skills are developed through years of training, thus the use of native forms typically costs more than standardized construction system.

This paper aims to make the above-mentioned qualities of materials with native forms more accessible by digital technologies. We use locally obtained branches which can be found almost everywhere not only in country-side but also in urban environment such as parks and along streets. Public service takes care of these branches: annual pruning, storing, and chipping or burning with some costs. The size of branches (from 5 -30 cm in diameter) is too small for furniture or other structural applications. It is a challenge for digital design and fabrication to utilize the diverse branches in meaningful ways. High-precision but low-cost scanning devices and personal digital fabrication machines make it possible to analyze and control natural materials with diverse native forms.

The key technical difficulty of materials with natives forms is design. Optimization is a straightforward approach: minimize an energy function which integrates structural and fabrication costs. This approach, however, is limited to particular design scenario with specific materials. Furthermore, the concept of optimum solution is well suited to goals such as efficiency or low-cost, but these goals are not the qualities materials with natives forms can compete with standardized construction systems. Instead, we take humans in our scan-design-fabricate workflow not only to solve the design problem, but also to provide an opportunity for people to participate in the workflow. Traditionally, in case of constructing public and symbolic buildings, such as church, people in a community took initiatives from fund raising to design, or even in construction process []. TODO: citation

In this paper, we report our case study to design and fabricate an architectural element out of irregularly shaped branches, using our humans-in-the-loop system. We developed an online platform where users can post branches found at hand, and design with them through the game *BranchConnect*. The game system itself helps users to design valid branch layouts and enable them to fabricate customized joineries to connect them together without screws and adhesives. The design of our joinery extends the traditional orthogonal lap-joints to various angles within a range, freeing the diverse forms of woods from orthogonal connections. Physically collected branches are digitally scanned and stored in a cloud database *BranchCollect*, and offline application *Branch Importer* analyzes forms of branches and upload to the database. The simple visual feedback and scoring system of the game guide users to valid solutions, which are further inspected by an offline application *G-Code Generator* for CNC milling process. The game system and developed import/export applications are currently limited to branches, however, the principle of human-in-the-loop for design is applicable for other kinds of materials with diverse irregular forms. We hope our method sheds lights on materials such as waste from demolition of buildings for various design applications.

In summary, our contributions are

- a workflow enabling to take natural materials with native forms as design components.
- an online game-based approach to participatory design-fabrication.
- an algorithm to generate customized non-orthogonal joineries which are fabricatable by a CNC router.

2 Related Work

3D printers and CNC routers made digital fabrication more accessible, and pre-fabricated customized building components are often used in buildings nowadays [Knaack et al. 2012]. According to the theory by Pye [1968], these components are processed from highly standardized material, thus its digital fabrication process is "workmanship with certainty"; a batch process of reading G-Code and strict execution of the code. On the other hand, as "workmanship with risks" in digital fabrication, interactive fabrication enables machines to pick up uncertain happenings and react on it [Willis et al. 2011]. Mueller and her colleagues developed interactive laser cutting, taking user inputs and recognizing placed objects in a fabrication scene [Mueller et al. 2012]. While their system interprets objects as simple platonic geometry, our work takes the native forms of branches, however, our work does not support interactivity in fabrication process. Crowdsourced Fabrication project took advantage of humans-in-the-loop in their fabrication system [Lafreniere et al. 2016]. On the other hand, our work puts emphasis on crowdsourced design as a socially networked fabrication. As a crowdsourced design system, Jerry et. al., developed a platform for light users to design trees and plants [Talton et al. 2009]. Our work also developed online collaborative design platform but directly linked to the real-world.

There are few works that take natural materials with native forms as design components. Schindler and his colleagues used digitally scanned wood branches and used them for furniture and interior design elements [Schindler et al. 2014]. Monier and colleagues virtually generated irregularly shaped branch-like components and explored designs of large scale structure [Monier et al. 2013]. Using larger shaped forked tree trunks, *Wood Barn* project designed and fabricated custom joineries to construct a truss-like structure[Mairs 2016]. *Smart Scrap* project digitally measured lime stone leftover slates from a quarry and digitally generated assembly pattern of slates [Greenberg et al. 2010]. In industry, recognition of irregularly shaped objects is essential for waste management. *ZenRobotics* developed a system that sorts construction and demolition waste by picking objects on a conveyor belt using robotic hands [Lukka et al. 2014]. For factory automation purpose, there is a system that recognizes irregularly shaped objects and sort them into a container [Sujan et al. 2000]. While these projects demonstrated the capability of digital fabrication processes to handle irregularly shaped materials, design process with native natural materials is still dependent on experts.

Cimerman discussed architectural design practices that took computer-mediated participatory (architectural) design [Cimerman 2000]. He mentioned three motivations of digital participatory design:

- Including stakeholders in creation of one's environment.
- Experimenting diverse design tastes from multiple point of views.
- Solving complex design tasks with full of diverse solutions.

Opening database of available local materials, people with various backgrounds can involve in design process, which could lower the cost of design fee with natural materials with native forms. For example, *Nano-Doc* took gamification approach to search valid nanoparticle designs against tumors out of infinite design space [Hauert et al.]. *DrawAFriend* has developed an online game to collect bigdata for drawing applications which assist humans with auto-stroke assistance [Limpaecher et al. 2013]. While these works developed games for collecting valuable data for solving medical or engineering problems, our game is served as a collaborative design platform, aiming to solve the socio-cultural issues in modern buildings: 1.generic design and 2.detached context.

3 Workflow

As illustrated in the left of Figure 1, our workflow starts from physically collecting branches. The collected branches are uploaded to cloud database by *Branch Importer*, and served to online game-based design application *BranchConnect*. The game is working on browsers and accessible from laptop computers and mobile devices. As in the right of 1, users can explore a global design with multiple branch layouts. Once the global design is fixed, designed layouts are further inspected by *G-Code Generator*, which generates customized joineries for CNC milling. After finishing the milling process, users physically assemble branches and complete the fabrication process. The pipeline of the workflow is illustrated in the Figure 2. In this section, we describe three steps in the pipeline: Digital Model Acquisition, the Game System, and Fabrication.

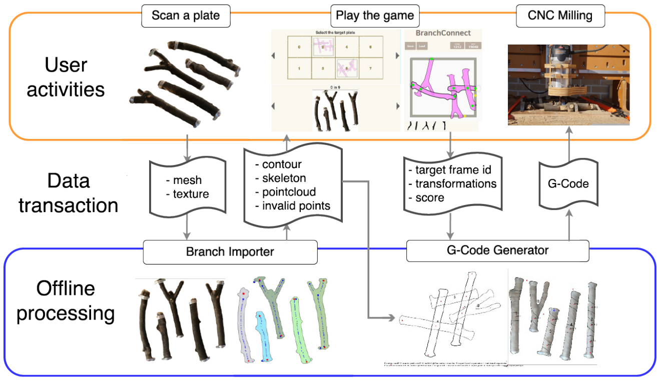

Figure 2: *A pipeline from model acquisition to fabrication.*

3.1 Digital Model Acquisition

Our system takes textured mesh model or point cloud with colored vertices. As complete mesh model provides more robust results with 3D shapes of branches, we describe our process based on mesh model as an input. There are various methods and software available for scanning 3D models. As for scanning setup, we describe in the Section 4. Taking mesh model with colored texture, our *Branch Importer* provides functions such as object detection, skeleton extraction, branch type classification, and fixture point setting.

The scanned result is a mesh model representing branches with a fixed plate. The system first idendifies branches by applying simple height threshold, and then applies contour detection (we currently use *findContour* in OpenCV [1]). The obtained 2D contours are used for extracting skeletons and clustering point cloud in the mesh model. Contours are triangulated and skeleton points are extracted from middle points on edges of triangles. These middle points are compared with top view image from OpenCV. If the point is inside of a contour, the middle point is counted as a valid point. After extracting valid middle points, the connectivity of skeletons is analyzed. In case grafting branch is detected, a new skeleton subbranch is added. The result is shown in Figure 3. Metal fixture locations are confirmed by simple mouse-clicks and set as invalid points. The acquired information is stored in a cloud database.

3.2 BranchConnect: The Game

The online game is accessible by laptops and mobile touch devices, and many users can play at the same time. The objective of the game is to collect valid layouts of branches which are fabricatable with 3 axis CNC milling machines. By analyzing the connectivity of branches and target points, the game checks structural feasibility of a given layout. The guidance system during furniture design inspected connectivity, durability, and stability [Umetani et al. 2012]. Unlike their work, our game focuses on fabricatability with minimum consideration of gemetric connectivity due to the limited resources with mobile devices.

[1]Open Source Computer Vision Library: http://opencv.org/

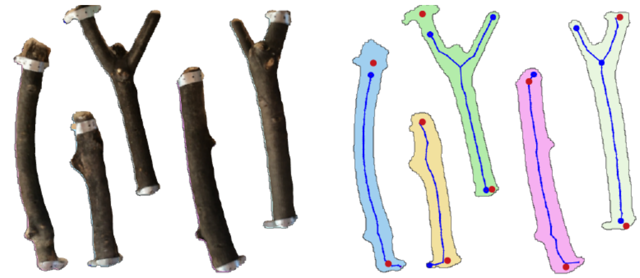

Figure 3: *An interface of Branch Importer. Left: a top ortho-view image of textured mesh model. Right: Extracted skeletons are shown with blue dots. The beginning of skeletons is shown bigger dots, and the red dots are invalid points defined by a user.*

Firstly a user selects a frame indicating multiple target points to be connected, and then selects a set of branches fixed on a plate (The left in Figure 4). After selecting the target frame and the set of branches, the user is guided to the game interface, consisting of the frame with the target points, and the set of available branches (The right in Figure 4). The user picks a branch from the available set on the bottom, and then places it in an arbitrary 2D pose through basic manipulations such as move, rotate, and mirror. The number of available branches differs depending on plates. With the feasible diameter of branches (over 2cm) and the plate size (50cm x 50cm), the number of available branches is most likely up to six. Within the limited number of branches, the user bridges all the target points by connecting all the used branches in one group. The game is completed when all the target points are connected. For higher score, the user can modify the design after the completion, and save it to the database.

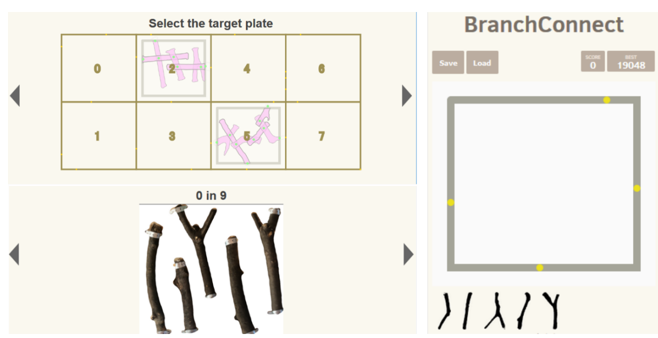

Figure 4: *Left: the selection interface for target frames (top) and branch panels (bottom). Right: the start interface of the game.*

System Overview

There are many collision detection libraries available, however, our game needs to detect intersected branch pairs, thus surface contact detection is overkill for our application. Also most branches come with free-form concave shapes, thus further geometric preparation such as convex decomposition is necessary for using these libraries. For fast and robust intersection detection, our system extensively uses skeletons of branches.

Hubbard and Philip developed collision detection by representing object with hierarchical 3D spheres aligned on skeletons [Hubbard 1996]. Our system takes similar approach but limited in 2D, but more focused on searching fabricatable joints. In the game, simplified skeletons are used to find the pair of closest skeleton points between two branches. When a branch is selected, it is counted as an active, and the system searches the closest skeleton point from skeletons of other branches. More precise joint calculation with higher resolution is further described in the next section.

A joint is created when an intersecting pair is detected, and the pair forms a group. The group is used for evaluating connections between target points (*Bridged*). The game is completed once all the target points are connected by a group of branches. The conditions of joint and group are indicated with simple color-code. Once the user finishes positioning, score is updated with weighted sum of parameters. Together with the color-code, the score update guides the user to form a valid design.

Joint Condition

Joint is the essential entity not only in the game but also in the fabrication process of customized lapped joineries. Each fabricated joint works as a rigid joint, and we do not calculate structural performance of each joint. Figure 5 illustrates valid and invalid joint conditions. Our joint only takes crossed pair (see Figure 5.1) because they are structurally stable, relatively simple to fabricate, and creates diverse designs. Tangential connections are counted as invalid as fabrication of tangential joinery is challenging with small branches (see Figure 5.3).

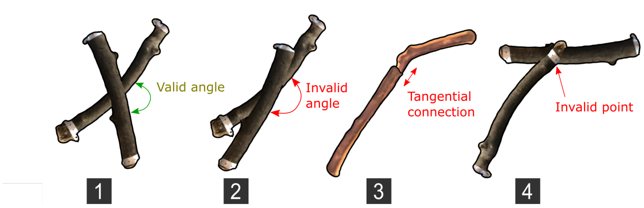

Figure 5: *Joint conditions. 1.valid joint. 2.invalid for violating the angle. 3.invalid tangential connection. 4.invalid for connecting on a fixture point.*

A valid joint's angle stays within a fixed range (see Figure 5.1 and 2). Joints close to metal fixtures are also counted as invalid (see Figure 5.4). Valid and invalid joints are displayed with green and red respectively. When a branch is connected to a target point, the specially added score is deisplayed in a pop-up square, also the graphics of target point's and the branch are changed. The branch is trimmed at the connection point with the target point, and the trimmed length is subtracted from the score.

To describe the process, we let $\mathcal{B}$ denotes a set of all the available branches, and each branch as $b_i \in \mathcal{B}$. While the process checks joint condition through all the branches $\mathcal{B}$, each detected joint is stored in each branch b_i, categorized in different conditions such as valid and invalid joints denoted as $j_{\text{valid},j,i} \in \mathcal{J}_{\text{valid},i}$ and $j_{\text{invalid},j,i} \in \mathcal{J}_{\text{invalid},i}$ respectively. When a branch b_i is connected to one of target point $t_j \in \mathcal{T}$, the t_j is stored in b_i.

A flowchart of the game system with joint and group conditions is illustrated in Figure 6.

Group Condition

After checking joint conditions of all the pairs of branches, the system checks the number of groups as well as its connection with the target points on a frame. If a group is not connected to any target point nor other groups, the group is *Islanded* and structurally invalid. While a user is positioning a branch by dragging or rotating,

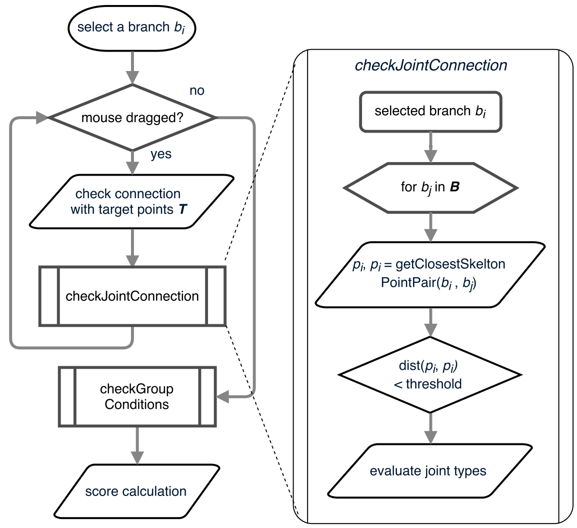

Figure 6: *Left: an overview of the game system with 1.joint checkand 2.group check and 3. score calculation. This process is iteratively executed while a user is exploring layout by dragging a branch. The joint check process is further illustrated in the right, and group condition check is described in Algorithm 1.*

groups are continuously calculated and indicated by simple color (Figure 7).

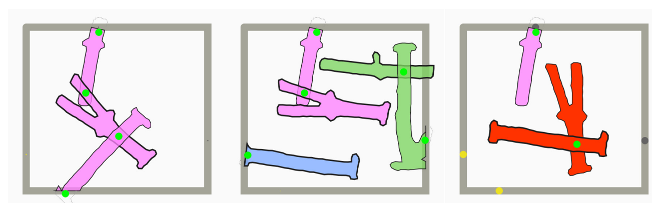

Figure 7: *Left: valid group with two target points connected. Middle: valid but three groups. Right: invalid due to the Islanded situation.*

After all the joint conditions are checked, we evaluate group conditions. Through checking the all the branches $\mathcal{B}$, the first group g_0 is created and stored as b_0. The game is completed when the number of $\mathcal{G}$ is one, and all the target points are connected with the group. The algorithm chekcing group conditions is described in 1.

Score Calculation

In the score calculation processes, follwoing entities forms the score: the numbers of valid and invalid joints on each branch, the number of groups as $N(\mathcal{G})$, the number of islanded groups as $N(g_{\text{islanded}} \in \mathcal{G})$, the number of bridged target points as $N(t_{\text{bridged},i}) \in \mathcal{T})$. The trimmed lengths of branches which are connected with target points are denoted as $trimmed(t_j, b_i)$ The score is weighted sum of these joint and group conditions, denoted in Equation (1). The weights 'w1...w5' are non-negative weight

Algorithm 1 Group Condition Update Algorithm

function UPDATEGROUPS($\mathcal{B}$)
    Reset all the groups $\mathcal{G}$
    Create new group g_0
    b_0 is added to g_0
    if b_0 has connected target point $t_i \in \mathcal{T}$ **then**
        g_0 sets t_i
    g_0 is added to $\mathcal{G}$
    for each branch b_i in $\mathcal{B}$ **do**
        GroupConnection $\leftarrow false$
        for each group g_j in $\mathcal{G}$ **do**
            for each branch b_j in g_j **do**
                if $b_{\text{paired},i} \in \mathcal{P}_\rangle$ has b_j **then**
                    b_i is added to g_j
                    GroupConnection $\leftarrow true$
                    if (b_i has t_i) and (g_j has t_j) **then**
                        Set g_j as *Bridged*
                    if g_j has no t_j **then**
                        Set g_j as *Islanded*
                    break
        if *GroupConnection* is *false* **then**
            create new group g_{new}
            b_i is added to g_{new}
            g_{new} is added to $\mathcal{G}$

coefficients pre-adjusted in advance by authors.

$$
\begin{aligned}
Score = {} & w_1 \sum_{1}^{N(\mathcal{B})} \sum_{1}^{N(\mathcal{J}_{\text{valid},i})} j_{\text{valid},j,i} && + w_2 \sum_{1}^{N(\mathcal{B})} \sum_{1}^{N(\mathcal{J}_{\text{invalid},i})} j_{\text{invalid},j,i} \\
& + w_3 \sum_{1}^{N(\mathcal{G})} g_{\text{islanded}} && + w_4 \sum_{1}^{N(\mathcal{T})} t_{\text{bridged}} \\
& + w_5 \sum_{1}^{N(\mathcal{T})} trimmed(t_j, b_i) \\
\text{s.t. } & w_j \geq 0 \ \forall j \in 1, \ldots, 5
\end{aligned}
\tag{1}
$$

3.3 Fabrication

After a design is selected for fabrication, the validity of the design is further inspected with a high-resolution model. The *G-Code Generator* was developed for fine-tuning the design by checking real-time feedback of updated joineries on branches with scanned orientations (see Figure 8). Users can change fabrication parameters such as offset ratio for the side cuts, milling bit diameter, overlapping ratio for defining the center cut depth, feed-rate, moving height and so forth. After confirming the fabrication settings and milling paths, it generates G-Code.

Some fabrication factors such as mirror and invalid points are already considered by *Branch Importer* and the game system. In this section, we describe the process of joinery generation. Each joinery geometry is parametrically modeled with: two plane surfaces on the sides of branches and one plane top surface (see Figure 9). The geometry creates rigid connection with the irregularly shaped sections of branches.

Similar to the joint searching process with skeletons, the *G-Code Generator* searches a set of four closest points from high-resolution contours, expecting that every intersected contour has four curves. After finding the four closest points, it trims two curves from each contour of branch. (two from intersecting branch and two from intersected branch) at each joint. The trimmed contours are transformed to the original scanned orientation and used for generating milling paths. Two curves from an intersected branch are used for side cuts milling paths, which are inwardly offset paths of the original branch contours. The center cuts are paths that are plaining the top surface of the joint. Height of center cuts is calculated with a diameter of cross-section at the center of a joint and the actual height information stored in point cloud. TODO: strange sentence

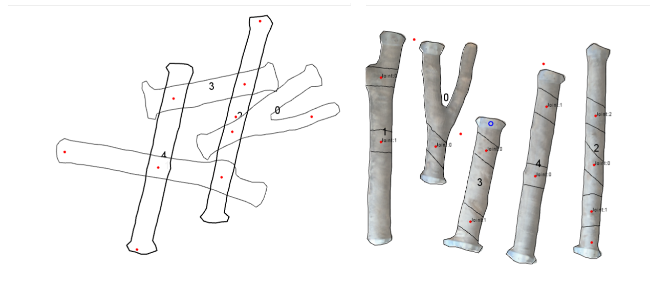

Figure 8: *Interface of G-Code Generator. Users can tweak the design on the left side and immediately see the updated joints and milling paths on the right.*

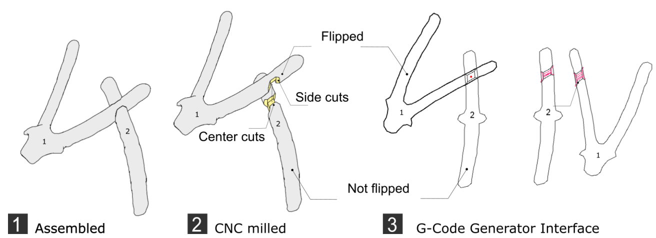

Figure 9: *An example of intersected pair: 1. an assembled pair of branches. 2. branches after the center and side cuts are milled 3. left: a layout defined by a user right: the original orientations of branches with generated milling paths with red color.*

4 Case Study

A design and fabrication workshop was organized to examine the validity of our system with a specific design target and a location. We selected a public community house where people in the community share the space and regularly use the facility and participants were selected among them. Participants are four children aged 10 and under (4, 7, 9 , and 10 years old) and two parents (Figure 10). We specifically selected children with the age range as non-experts without experiences in computational design or digital fabrication, also for observing the clarity and attractiveness of the game.

The goal for the participants is to contribute to an ongoing design and fabrication process of screen wall (2m by 0.9m) consisting of 8 rectangles. Six frames were already designed and built, thus the rest two frames were prioritized for them to design and fabricate in this workshop.

Participants are asked to follow the entire process from collecting and fixing the branches on a plate, scanning the plate, complete designs by playing the game, and assembly after the CNC milling. Participants were informed about the goal of the workshop, and each process was introduced by experienced tutors.

Figure 10: *An overview of the workshop. 1.the overview of the space. 2.collect branches. 3.cut 4.attach on a plate 5.scan 6.play the game 7.CNC milling.*

4.1 Setup

System and Hardware Setup

We used two iPad minis with iSense scanners attached for scanning branches, and a 3 axis CNC milling router with a 6 mm diameter milling bit. We used a laptop PC (Lenovo w240 with intel core i7) for running *Branch Importer* and *G-Code generator*, as well as operating the milling machine. The scan area of iSense is 500mm x 500mm, and the milling machine's stroke along z-axis is 70mm, which provide geometric constraints for available branch sizes. *BranchConnect* was hosted at *Heroku* cloud server [2], and we used *MongoDB* [3] as a cloud database.

Preparation of Branches

The participants were asked to collect branches with 2 - 10 cm in diameter. The lower bound of the diameter is for milling process would not destroy them, and the upper bound is for the limit of z-stroke of the CNC router. The collected branches are cut in arbitrary lengths, not longer than 500 mm due to the limit of scanning area.

As our game system and fabrication process take 3D branch shapes as 2D contours with height map, these constraints work positive for the system automatically filtering out branches with large 3D twists. We ask participants to scan with iPad + iSense and prepare feasible mesh model by themselves. After obtaining mesh models, tutors import models from iPads to a laptop and upload them to database by *Branch Importer*.

User Experiences with the System

After models were uploaded to the server, users can see that their plates are added in the selectable branch plates with their names and locations. Users can access to the start page by PC and mobile devices. We prepare both options and let participants choose a device.

A user is firstly directed to a start page and asked to submit a user name. Secondly, the user is navigated to target frame selection page, and asked to pick one out of eight frames. Each frame has different target points. The interface also shows the completed branch organizations within each target frame. If there are multiple designs, three designs with highest scores are displayed. The user can change the currently displayed design by clicking within each frame and choose either starting their design from scratch, or select the design and improve it.

[2]Heroku is a platform as a service (PaaS) that enables developers to build, run, and operate applications entirely in the cloud. https://www.heroku.com/

[3]MongoDB is a free and open-source cross-platform document-oriented database program. https://www.mongodb.com/

After selecting a target frame, the user goes to branch selection page, displaying 15 plates when the workshop was held. In this page, they see the plates made by themselves on the page, as well as their names on the plate. The user can select the same plate for designing other target frames. By clicking a displayed branch plate, the user is navigated to the game interface. After completing to bridge all the target points, the design is automatically uploaded to the database, but the use can continue to design. Tutors and participants select design solution for each target frame, and an experienced tutor operates *G-Code Generator* as well as the CNC router. Participants were asked to assembly branches after joineries were milled.

4.2 Results

The entire workshop took 4.6 hours to complete the whole process, including introduction, moving, and pauses. Table 4.2 shows durations of each task.

Task	Duration (hour)	Fraction (%)
Introduction	0.3	6.5
Collecting branches	0.6	13.0
Preparing plates	0.8	17.4
Preparing models	0.3	6.5
Uploading models	0.2	4.3
Designing by the game	0.5	10.8
Inspecting models	0.2	4.3
CNC milling	0.5	10.8
Assembling	0.2	4.3
Miscellaneous	1.0	21.7
In total	4.6	100

Collecting Branches

The diameter and length constraints for available branches worked as guidelines for participants rather than restricting finding and cutting arbitrary branches. After cutting branches in certain lengths, participants fixed branches on plates by thin metal plates with screw holes. It was straightforward for them to firmly fix branches so that they are not moved during milling process. These fixture points are counted as invalid points in the game where joinery points can not be generated. The participants built two plates with three and five branches fixed on each plate.

Model Acquisition

iSense 3D scanners come with an intuitive software for scanning and modifying models. After we gave an instruction, most of participants practiced several scans and successfully scanned models without any problem.

Each scanning and re-touching took 2-3 minutes, and 30 seconds for generating data by *Branch Importer*. Including the prepared panels previously, we scanned 15 plates in total, 75 branches, and 35.3m of total length including sub-branches. TODO: check the length again! We got 40 I shaped branches, 19 V shaped branches, and 16 Y shaped branches. The result is shown in Figure 11.

Design with the Game

Most participants used iPads for navigating pages and playing the game. All the participants understood the goal immediately, however, they had difficulties with mobile touch interface, such as rotation and flipping operation by gestures.

One participant switched to play by a PC for more precise control due to the problem. All the participants chose to develop their own

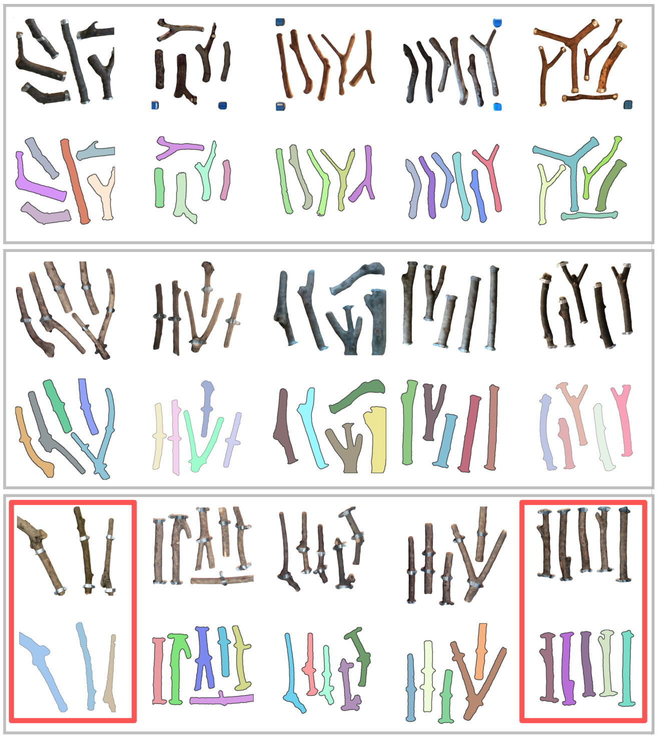

Figure 11: *An overview of all the 15 scanned plates for the workshop. Top raw of each set shows ortho-top views of scanned mesh models, and the bottom raw is the recognized branches assigned random colors. The red-lined rectangles indicate the plates built by participants in the workshop.*

designs from the scratch, although they had instruction about the ”continue existing designs”.

Although there is no time limit in the game system, we set 30 minutes for playing the game, and 8 solutions were given by participants. xxx frames were completed per participants and xxx participants completed the whole eight target frames. The average score is xxx, and average playing duration was xxx to complete each target frame.

Overall Design Consensus TODO: this section might be removed

As the target frame selection page can display designs not only from one user but also from all the others at once, we could get an overview of design options. The designs are displayed as score descending order but limited numbers, we could find feasible solutions with mostly all the target points were bridged. As participants were excited by seeing their branches and designs, we took some of their solutions and their plates for the fabrication although their solutions did not satisfied completion of the game.

Fabrication

We did not have major problems for converting designs to G-Code milling paths as we encountered major issues regarding the fabrication before the workshop, which are reported in this section.

The main problem was the accuracy of acquired contours. We observed most of scanned models had occluded regions between plates and branches, which create interpolated faces during solidifying process, resulting in outwardly offsetted contours. TODO: check offsetted correct english After milling was finished and when branches were assembled, six pairs of branches were loosely connected because the calculated contours were 2-3mm eroded than the actual sizes. We avoided this problem by trimming branches from 2-5 mm higher than the plate surface. After this operation, the rest of connections were tightly connected.

We also observed that many milling paths were 5-10 mm off from the center of planned joints. Multiple reasons could be considered as reasons such as,

- deformation of mechanical parts of the CNC router
- not dense resolution of acquired contours of branches
- misaligned orientation of the plate compared to the scanned model

To avoid the misalignments, we modified the *G-Code Generator* so that an operator can freely adjust the absolute origin of the generated milling paths. The origin was usually set with around the center of the plate. After this modification, the misalignment from joint center was reduced with 5 mm off at the maximum misalignment. Branches could absorb 3-5 mm misaligned joint positions due to the elasticity of the wooden materials. The misaligned joint positions worked as post-tensions, solidifying the structure. We assume that this is only applicable when an applied bending moment and cut surface at a joint is orthogonal or not too much off from orthogonal. TODO: this sentence

4.3 Summary

The objective of the workshop was to observe the validity of our system for non-expert users. We could see that participants aged 10 and under were capable of following the workflow. Participants were encouraged to contribute to the ongoing design with the online platform, and successfully provided designs from available branch plates.

We had several requests from participants regarding the game interface but also related to the workshop organization. Several participants requested to allow multiple branch plates for designing a frame, or even remove the target frame and let them freely design with branches. Also a participant who gave up with iPad operations requested additional buttons for mobile touch interface to keep an active branch selected.

A noteworthy fact was that one participant with four years old failed to complete the game, but insisted on accepting his design to be fabricated by the CNC router. We took these requests as their commitment to the entire workflow, as we observed that all the participants insisted on continuing the fabrication process.

5 Conclusion

Summary

In this paper, we presented a workflow to design and fabricate with branches which are not large enough for producing standardized building components. Our workflow was validated with case-study with lower-aged participants without design and fabrication experiences. Our online platform to store geometries of scanned branches and compositions of branches was reachable by people with various backgrounds. Our light-weight branch joint detection algorithm was also validated by running on the browser game, contributed to the accessibility of the presented workflow. Together with the accessibility, the intuitive interface was simple enough for non-expert users. We successfully built a network of branches with rigid joints generated by our joinery milling path generator. Each of joints has customized lapped-joint geometry, which is challenging to design and fabricate.

Limitations and Future Work

Our system is limited to handle 3D shaped branches as 2D with height information only used for segmentation and G-Code generation. Our workflow requires branches to be fixed on a plate, which takes the longest duration in the workflow except for moving and pause in between tasks. Despite of successfully fabricated non-orthogonal joineries, we did not complete the attaching branches to target frames, as we prioritized to validate branch-branch joineries.

Our design workflow fully relies on human's ability for obtaining design solutions. As the problem of limited number of branches and fixed target points to be connected, the system could assist humans to reach to structurally sounding solutions with less efforts. Collected data from game could be further analyzed for extracting meaningful data for the development of assisting algorithm. Our joint and group detection, and milling path algorithms are not applicable to other purposes at this moment. It is valuable to compare the performance of our joint and group detection system to other collision detection library. In our workflow, the pipeline is not seamlessly connected: experienced operators need to take care of retouching mesh model, Branch Importer, and G-Code generator, and operating a CNC router.

Acknowledgements

We appreciate the public house for hosting the CNC router and providing participants for the user study. We thank to the film editor, Shin Yamane for shooting our video clips. We also thank to the developer of an online game *2048*, Gabriele Cirulli and his contributors, for sharing source code in Github community.

References

Cimerman, B. 2000. Participatory design in architecture: can computers help? In *PDC*, 40–48.

Greenberg, B., Hittler, G., and Perry, K., 2010. Smart scrap. Accessed: 2016-09-30.

Hauert, S., Lo, J. H., Nachum, O., Warren, A. D., and Bhatia, S. N. Crowdsourcing swarm control of nanobots for cancer applications.

Hubbard, P. M. 1996. Approximating polyhedra with spheres for time-critical collision detection. *ACM Trans. Graph. 15*, 3 (July), 179–210.

Khoshnevis, B. 2004. Automated construction by contour crafting related robotics and information technologies. *Automation in construction 13*, 1, 5–19.

Knaack, U., Chung-Klatte, S., and Hasselbach, R. 2012. *Prefabricated systems: Principles of construction*. Walter de Gruyter.

Lafreniere, B., Grossman, T., Anderson, F., Matejka, J., Kerrick, H., Nagy, D., Vasey, L., Atherton, E., Beirne, N., Coelho, M. H., et al. 2016. Crowdsourced fabrication. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, ACM, 15–28.

Limpaecher, A., Feltman, N., Treuille, A., and Cohen, M. 2013. Real-time drawing assistance through crowdsourcing. *ACM Transactions on Graphics (TOG) 32*, 4, 54.

Lukka, T. J., Tossavainen, T., Kujala, J. V., and Raiko, T. 2014. Zenrobotics recycler–robotic sorting using machine learning. In *Proceedings of the International Conference on Sensor-Based Sorting (SBS)*.

Mairs, J., 2016. AA design and make students use a robotic arm to build a woodland barn. Accessed: 2016-09-30.

Monier, V., Bignon, J. C., and Duchanois, G. 2013. Use of irregular wood components to design non-standard structures. In *Advanced Materials Research*, vol. 671, Trans Tech Publ, 2337–2343.

Mueller, S., Lopes, P., and Baudisch, P. 2012. Interactive construction: interactive fabrication of functional mechanical devices. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, ACM, New York, NY, USA, UIST '12, 599–606.

Oliver, P. 1997. *Encyclopedia of vernacular architecture of the world*. Cambridge University Press.

Pye, D. 1968. *The nature and art of workmanship*. Cambridge UP.

Schindler, C., Tamke, M., Tabatabai, A., Bereuter, M., and Yoshida, H. 2014. Processing branches: Reactivating the performativity of natural wooden form with contemporary information technology. *International Journal of Architectural Computing 12*, 2, 101–115.

Sujan, V., Dubowsky, S., Ohkami, Y., et al. 2000. Design and implementation of a robot assisted crucible charging system. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2, IEEE, 1969–1975.

Talton, J. O., Gibson, D., Yang, L., Hanrahan, P., and Koltun, V. 2009. Exploratory modeling with collaborative design spaces. *ACM Transactions on Graphics-TOG 28*, 5, 167.

Umetani, N., Igarashi, T., and Mitra, N. J. 2012. Guided exploration of physically valid shapes for furniture design. *ACM Trans. Graph. 31*, 4, 86–1.

Weston, R. 2003. *Materials, form and architecture*. Yale University Press.

Willis, K. D., Xu, C., Wu, K.-J., Levin, G., and Gross, M. D. 2011. Interactive fabrication: new interfaces for digital fabrication. In *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*, ACM, 69–72.