芸術科学会論文誌 Vol. 21, No. 2, pp. 65-76 (2022)

RePoKaScope: 立体万華鏡のためのデザイン支援システム

本間梨々花[1)]　越後宏紀[2)]　五十嵐悠紀[1),3)]（正会員）

1）明治大学総合数理学部　　2）明治大学大学院先端数理科学研究科

3）お茶の水女子大学理学部情報科学科

RePoKaScope: A Design Support System for Stereoscopic Kaleidoscopes

Ririka Honma[1)]　**Hiroki Echigo**[2)]　**Yuki Igarashi**[1),3)](Member)

1) Department of Interdisciplinary Mathematical Sciences, Meiji University

2) Graduate School of Advanced Mathematical Sciences, Meiji University

3) Department of Computer Science, Ochanomizu University

{ev180563, hiroki_e} @ meiji.ac.jp, yukim@acm.org

アブストラクト

立体万華鏡とは，立体を形成する面が鏡面であり，模様が三次元空間に広がるタイプの万華鏡である．鏡面に模様をデザインする際，鏡面反射を繰り返して表示される三次元空間の模様を想像することは困難である．また，鏡面を削って制作を行うため，納得のいくデザインを完成させるまでに試作品を何度も制作する必要があった．この問題を解決するために，本稿では立体の中でも正多面体に注目し，正多面体から成る立体万華鏡の模様のデザインを支援するシステムを提案する．三次元空間に映し出される模様をシステム内でシミュレートすることで，試行錯誤をシステム内で完結し，ユーザが満足するデザインができてから立体万華鏡を制作することが可能になる．

Abstract

In the production of stereoscopic kaleidoscopes, it is difficult to imagine the patterns in three-dimensional space displayed by repeated specular reflection. In this paper, we propose a system to support the design of patterns for a Stereoscopic Kaleidoscopes consisting of regular polyhedra. The user can design original pattern by simulating the pattern in the system.

キーワード 立体万華鏡，デザイン，シミュレーション，制作

Keywords Stereoscopic kaleidoscope, Design, Simulation, Fabrication

1. はじめに

万華鏡とは，複数の鏡を組み合わせて筒を作り，筒状の中に入っているものの美しい模様を映し出す工芸品である．小さい子から大人まで親しまれている万華鏡は，制作キットなどを利用することで初心者でも簡単に手作りすることができ，同じ万華鏡でも回転させることで映し出される模様を楽しむことができる(図 1)．このような万華鏡は 1816 年に David Brewster によって考案された平面充填する万華鏡であり，これを満たす三角形の構造は 3 種類しかないことが示されている[1,2].

これに対して，立体万華鏡というものが存在する．ヤマザキミノリ氏によって 1974 年に考案された立体万華鏡 CUMOS[3]は空間充填の万華鏡であり，空間を埋め尽くすような幾何学模様が描かれることが特徴である．立体万華鏡は，ワークショップを中心に制作することが可能だが，鏡で反射した完成デザインをあらかじめ想像しつつ，カッターで鏡を削って制作しなけれ

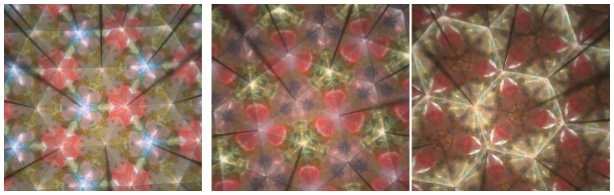


図1　同じ万華鏡でも回転により映し出される模様は変化する.

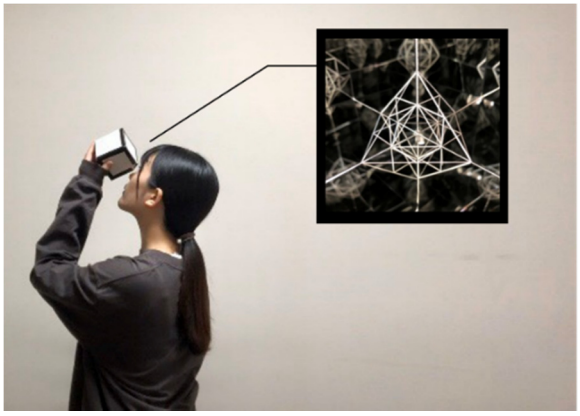


図2　本システムを用いてデザインした立体万華鏡.

ばならず，制作初心者にとって想像通りのデザインを制作することは困難である．また，鏡を削って制作するため，一度デザインしたものを途中で変更することは困難である.

この問題を解決するために，本稿では，立体万華鏡に対する模様のシミュレーションを行うことができるデザインシステム(図 2)を提案する[4]．本システムでデザイン可能な立体として正多面体の5種類すべてを対象とした．立方体は空間充填であり，その他の4種類の正多面体は空間充填ではないが，これら全てに対応可能な描画アルゴリズムを実装した．本システムのユーザ実験を行い，本システムの有用性について議論する.

2. 関連研究

万華鏡のような幾何学模様を描くソフトウェアは多く存在する．万華鏡タイル[5]は，万華鏡の原理を使って，平らな多角形で構成された多面体や，滑らかな球面を対象としてタイル貼りしたものを描くことができる．ibisPaint X [6]は，回転式の対称定規を使って万華鏡のような幾何学模様を簡単に作成することができる．万華鏡カメラ PrismScope[7]では，カメラで撮影された映像をリアルタイムで画面に万華鏡のように映し出すことができる．東藤ら[8] は Web カメラで撮影された映像をリアルタイムで処理を行い，万華鏡のように表示するシステムを実現した．また，望月のデジタル万華鏡[9]は，フラクタル画像解析を芸術分野に利用している．本稿では，立体万華鏡を制作するにあたり，1 画面でデザインした模様を，リアルタイムに万華鏡表示を行うことで，ユーザの希望に沿ったデザインが行える．また，万華鏡表示の精度を 2 段階用意し，ユーザが選択可能にすることで，ユーザが試行錯誤する支援を目指した.

工作を題材とした研究として，ビーズ[10]や編み物[11]などの設計および制作工程を支援するものが提案されている．これらは，制作工程を間違えたときに後戻りが可能である一方で，立体万華鏡は鏡面に一度傷を付けると修正ができないという特徴がある．このように途中で修正が困難である特徴を有するものとして，木目込み細工や紙巻きオルゴール漫画があげられる．木目込み細工の設計支援システム[12]では，木目込み細工を行った再現画像を確認しながらスケッチインタフェースを用いてデザインを描くことで，完成系をイメージしながらデザイン制作を行うことができる．紙巻きオルゴール漫画の制作支援システム PomPom[13]では，イラストと作曲したメロディをシステム内で確認しながらデザイン制作を行うことができる．本システムでもシステム内であらかじめシミュレーションを適用することで，ユーザが満足したデザインになったことを確認し，その後に鏡を削り実際に制作していくという方針でユーザの立体万華鏡設計を支援していく.

3. 立体万華鏡の作り方

立体万華鏡は，立方体の立体万華鏡が主流であり，ワークショップを中心にラビリンスボックス[14]と呼ばれている(以下，立方体の立体万華鏡を「ラビリンスボックス」という)．しかし，立体万華鏡は立方体のみではなく，正多面体での制作が可能であるため，本稿で提案するシステムでは正多面体も含めた制作支援を目指した．本章ではまず，実際の立体万華鏡の制作工程について，ラビリンスボックス，正多面体の順で述べる.

3.1 ラビリンスボックスの制作工程

ラビリンスボックスを制作する過程は図3に示したように以下の工程からなる；

① ポリカーボネイトミラー板（以下，ミラー板という．)を6枚(面の数)用意する.
② 6枚のうち3枚にペンを用いてデザインの下書きをする.
③ ②で下書きをした部分を，アクリルカッターで削る.
④ ②，③で使用しなかった3枚のミラー板の角を切断する．(覗き穴となる．)
⑤ アクリルカッターの背を用いて鏡面のバリを取り除く.
⑥ ③で使用した3枚を透明なテープで貼り合わせる.
⑦ ④で使用した3枚を不透明なテープで貼り合わせる.
⑧ ⑥，⑦で余った辺を貼り合わせ，組み立てる.
⑨ ミラー板の鏡面についている保護シートを剥がす.
⑩ ⑧を不透明ビニールテープで組み合わせて完成.

また，③のミラー板に模様を描く工程では，図 4 のように，除光液をつけた綿棒で広範囲を削ることもでき，カッターでは難しい図形の塗りつぶしや太めの波線などを描くことができる(図 5)．そのため，塗りつぶしたい箇所は綿棒の使用を推奨する.

ラビリンスボックスは，利用するミラー板 6 枚のうち，3 枚に同じ模様をデザインすることが一般的である．ミラー板によって違う模様をデザインすることも可能であるが，3 枚同じ模

図3　ラビリンスボックスを制作する過程.

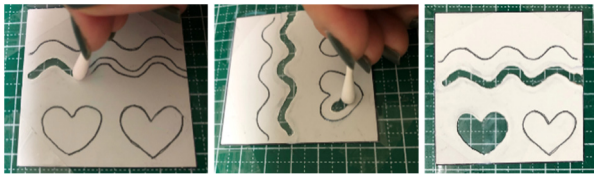


図4　綿棒で削る様子.

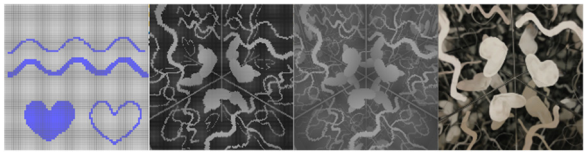


図5　綿棒を使った制作結果.

様にした方がミラー板によって鏡面反射して映し出された模様が綺麗に見ることでき，これもラビリンスボックスの特徴であると考える．そのため本研究では，3枚とも同じ模様をデザインするための制作支援システムを開発する.

3.2 正多面体の立体万華鏡の制作工程

正多面体の立体万華鏡は，3.1節で述べたラビリンスボックスの制作過程と同様の過程で制作でき，ミラー板の面の数を変更するのみである．ただし，正四面体の場合，④の覗き穴を作成する工程において，複数のミラー板の角を切断するのではなく，鏡面をくり抜く形で覗き穴を作成する(図6)．これは，覗き穴から万華鏡を覗いた際，注視点となる箇所が頂点になるようにすることで，綺麗な模様が映し出されるためである．正八面体，正十二面体，正二十面体は，ラビリンスボックスと同様に複数のミラー板の角を切断することで覗き穴を作成する．この覗き穴と注視点の関係については，5章で述べることとする.

我々は，この制作工程のうち，②，③に着目した．この制作工程では，ユーザが好みの模様をデザインするため，最も試行錯誤が必要な部分であるが，③の工程でミラー板に傷を1度つけてしまうと修正することができない．また，模様をデザインしている間は，鏡面反射した完成予想図を想像しながら制作しなければならず，初心者にとっては制作が困難である．そのため，ユーザが理想の立体万華鏡を制作できる支援として，本システムでは②の模様をデザインする工程をソフトウェア上で行うとともに，鏡面反射した完成予想図をシステム内で提示する.

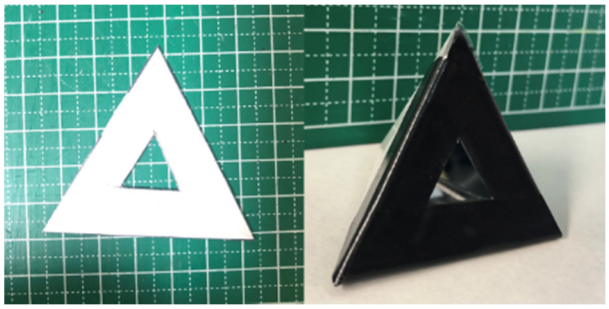


図6　正四面体の覗き穴.

4. 提案システム RePoKaScope

本章では，提案システム RePoKaScope を用いたデザイン制作の流れ，正多面体のデザインについておよびユーザインタフェースについて述べる.

4.1 提案システムを使ったデザイン制作の流れ

本システムは Processing を用いて実装した．対象とするユーザは初心者であり，GPU を使わずに一般家庭で普及しているノート PC を使い，リアルタイムでデザインできることを重視している．一連の流れを図 7 に示す.

ユーザは，まず方眼上のキャンバスにドット絵を描くように模様のデザインを作成する(図 7(a))．キャンバス画面とシミュレーション画面が分かれて存在しており，シミュレーション画面を見ながらキャンバスで模様のデザインを作成することが可能である.

次に，シミュレーションを行うことで，完成予想図を視覚で確認する．シミュレーションは2種類に分かれており，数秒から数十秒で確認できる簡易的なシミュレーション(図7(b))と，数分で確認できる実物に近い詳細なシミュレーション(図7(c))がある．シミュレーション結果を表示することで，鏡面で反射したデザインを視覚で確認することができ，完成形を想像しながらデザインすることが可能である． 本システムではキャンバス画面の下部に「速い」「きれい」という選択式のボタンを設置し，キャンバス画面からユーザが好んだシミュレーション結果を表示することができるように設計した．シミュレーションのアルゴリズムについては5章，シミュレーションのレンダリング時間については6.1節で述べる.

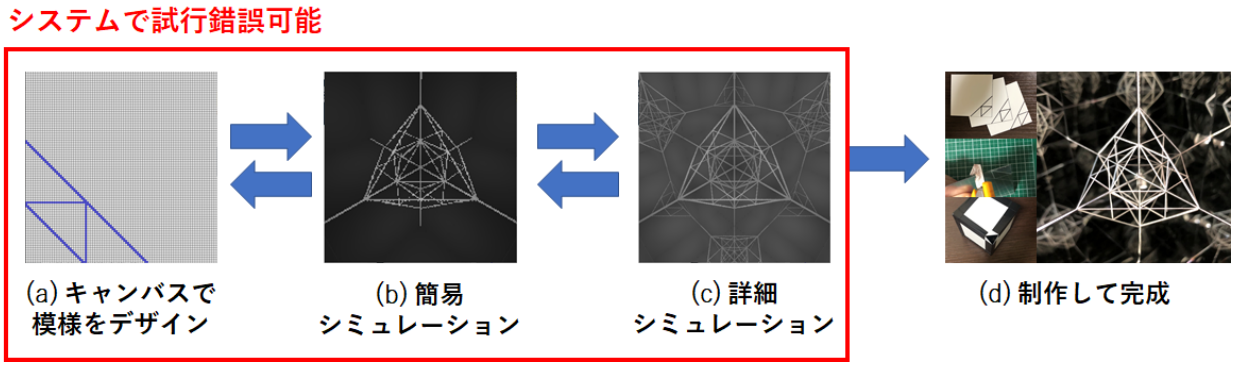


図7　デザイン制作の流れ.

4.2 正多面体のデザイン

本システムでは，図 7 のような立方体のほかに，正四面体，正八面体，正十二面体，正二十面体の全正多面体 5 種類の立体万華鏡のデザインを行うことができる．システムで任意の正多面体を選択でき(図 8)，デザインの途中で他の正多面体に変更することも可能である．キャンバス画面では，多面体によって面形状が異なるため，キャンバスの形とキャンバスに敷き詰める図形を変えている．正四面体，正八面体，正二十面体の面形状は正三角形であるため，キャンバスは正三角形を敷き詰めて構成されている(図 9)．一方，正十二面体の面形状は正五角形であり，方眼のようにキャンバスに敷き詰めることが不可能であるため，ひし形をベースに対角線を引いた三角形で対応している(図 10)．キャンバスのマス目の細かさ（敷き詰めた図形の大きさ）についてはユーザ実験を行い，デフォルト値を決定した．この値は実際の制作物の大きさに応じて適切な値にユーザが変更することもできる．詳細については 6 章に述べる．

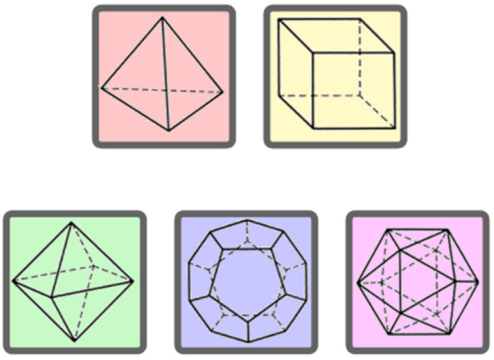


図 8　選択画面.

4.3 提案システムのユーザインタフェース

デザインを行いやすくする工夫としては，(1)デザイン時に関する注意事項の提示(図 11)，(2)塗りつぶし機能，(3)戻る機能の 3 点が挙げられる．

デザイン時の注意点は 3 つあり，「塗りつぶす面積を小さめかつ少なめにすること」，「鋭角を含む塗りつぶしを避けること」，「曲線を描く場合は，なるべく太めにすること」である．1 つ目に関しては，削る面積が大きいほど，万華鏡外部がよく見えることで，模様の見え方に影響を及ぼす可能性があるためである．2 つ目は，図 4 で示したように，塗りつぶしは綿棒を使用することから，細い部分を正確に削ることが困難であるためである．3 つ目は，カッターで波線を描くことは難しいので，綿棒の使用を推奨し，綿棒で正確に削れるほどの幅を取るためである．カッターの使用に抵抗がない方は気にしなくてもよい．これらの注意点を確認した後，デザインを開始する設計とした．

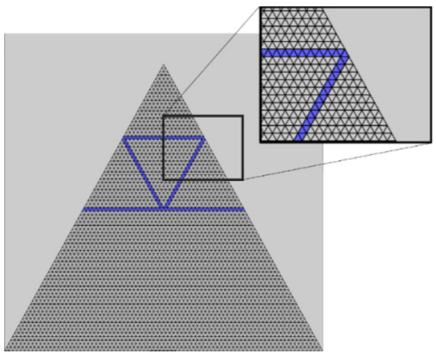


図 9　正三角形のキャンバス.

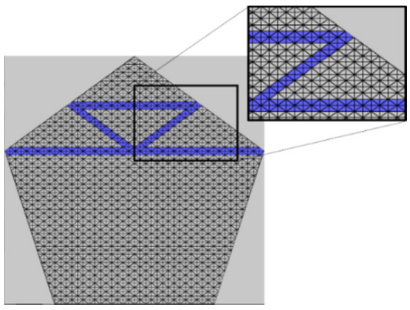


図 10　正五角形のキャンバス.

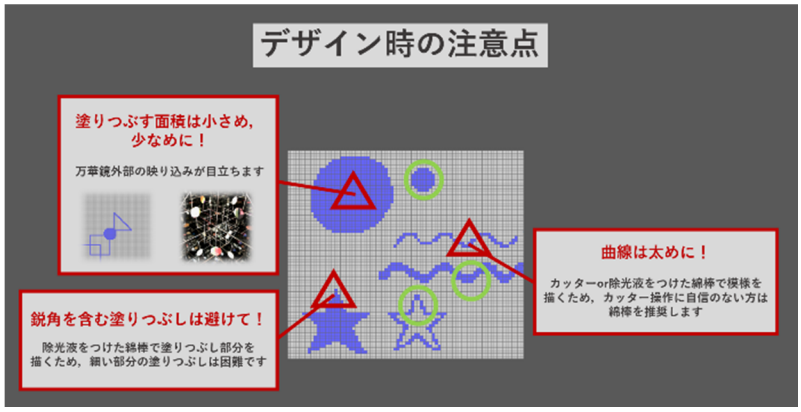


図11　デザイン時の注意点.

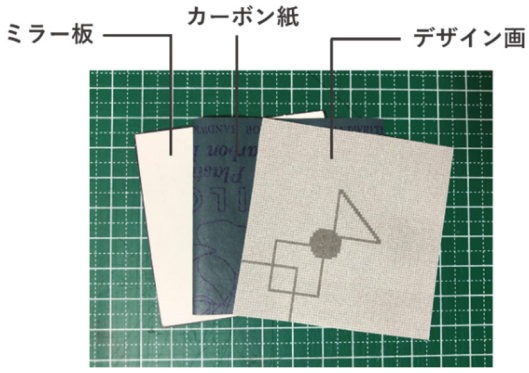


図 12　印刷したデザイン画を使用した下書き.

塗りつぶし機能は，Flood fill アルゴリズムを基に実装しており，青い領域で囲まれた内側を選択することで，範囲内のすべてを青く塗りつぶすことができる．これは方眼のキャンバスだけではなく，正三角形，正五角形のキャンバスにおいても同様に塗りつぶし機能を実装している．UNDO 機能を用意することで，塗りつぶしを行う際にうまく囲みきれず，広範囲が塗りつぶされてしまった場合に有効である．

デザインが完成したら，「保存」ボタンを押下することでデザイン画が保存される．デザイン通りの模様を生み出すため，デザイン画を印刷し，3.1 節の制作工程②を行う際，図 12 のようにミラー板，カーボン紙，印刷したデザイン画と重ねて下書きを行うことが可能である．

5. アルゴリズム

本システムは，鏡面のデザインを行うキャンバスと，シミュレーション結果を表示する 2 つのウィンドウから成る．本章では，模様のシミュレーション方法とリアルタイム性向上のための工夫について述べる．

5.1 模様のシミュレーション方法

本システムは，レイトレーシング を用いて完全鏡面反射を模倣することで，模様のシミュレーションを行った．まず，正多面体の各面を完全鏡面反射するものとした．鏡面に付ける模様は，描いたデザインに合わせて，数多の三角形を鏡面から立体の内側に向けて，微小な値を離した位置に配置することで表現している．また実物では，鏡面に付けた傷から光が入ることで，立体内部の明るさが保たれているが，本システムでは，点光源を立体の中心に配置することで対応している．そのため，ユーザが描いた線に位置している図形(キャンバスを敷き詰めている図形のうち，塗りつぶされたもの)を万華鏡内部に配置することで，デザインした模様を表現している．さらに，模様を付ける面同士が接する辺から生まれる光軸は，あらかじめ実物で光が入り込むための隙間(図 3 の⑥で作る隙間)と同じ大きさのオブジェクトを配置することで表現している．これらに伴い，付影処理の実装は行っていない．

鏡面反射によって映し出された模様は，再帰回数の回数ごとに色を薄く表示している．そうすることで，奥行きを表現することができ，より現実の完成予想図に近い見た目を再現している．

視点と注視点の位置を図 13 のように定めた．すべての正多面体に対して，模様を付けるすべての面が接する点を注視点とし，この注視点と立体の中心を結ぶ直線が注視点以外で再び立体と交わる点を視点とした．視点は，立体万華鏡の覗き穴となる．

模様の位置は，キャンバスに敷き詰めている各々の図形に 2 次元配列を与えることで保存している．また， 1 つの万華鏡において全面同じ模様であることを前提としているため，1 面目以外は，注視点を回転の中心として 1 面目の配置座標を回転移動させた．

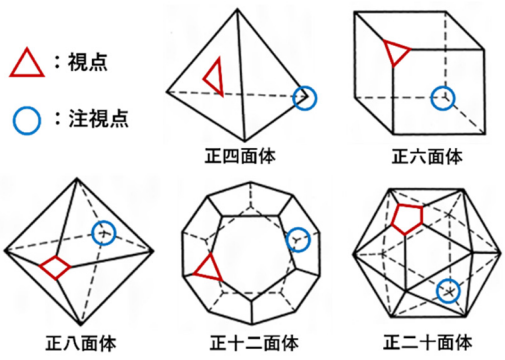


図 13　各正多面体の視点と注視点.

5.2 リアルタイム性向上のための工夫

立体万華鏡は，完全鏡面反射する物体が向かい合わせとなる構造でもあり，レイトレーシングを行う関数の再帰回数を設定する必要がある．一方で，本システムでは，模様をデザインする制作工程において，容易に試行錯誤が可能であるようにするため，シミュレーションに要する時間を短縮することが求められる．そのため，我々はレンダリングを 2 種類用意した．1 種類目は，リアルタイム性の追求を目的とし，短時間でシミュレーション結果を提示する．2 種類目は，1 種類目より詳細で正確なシミュレーション結果を提示することを目的とし，ユーザの最終的な意思決定に役立つことを期待する．具体的には，1 種類目では，再帰回数を 1 回に設定するとともに，2 種類目で設定している参照数の 1/9 に減らしている．そのため，シミュレーション結果は粗く提示されるものの，鏡面反射の概観を把握でき，レンダリング時間を数秒から数十秒でシミュレーションすることを可能にしている．2 種類目では，再帰回数を 4 回に設定しており，正確なシミュレーション結果を提示可能なものの，レンダリング時間に数分程度を要す．この 2 種類のレンダリング時間については次章で詳しく述べる．

また，マウスクリック操作が行われた後にシミュレーションが行われるようにすることで，シーンに置く物体を保管するリストに必要以上の物体が追加されることを避けた．1 種類目のシミュレーションを繰り返す場合でも，余計な計算を省くため，1 度シミュレーションを行ったデータを保持し，新しくキャンバスで描かれた模様に位置する図形のみを新たにリストに追加するようにした．

6. 模様デザイン制作時のキャンバス設計

模様をデザインする際，GPU を使わないノート PC を利用して短時間でシミュレーション結果を表示することを本システムは目指しているため，白紙で自由に描画するキャンバスではなく，正三角形や正方形を敷き詰めたグリッド状のキャンバスを用意することとした．このキャンバスに敷き詰める図形の大きさを決めるにあたり，図形の大きさごとに，同じ模様(図 14)を描いたときにかかるレンダリング時間を比較した．また，立体万華鏡の制作経験がない 20 - 25 歳の男女 5 名に，各キャンバスにおける模様の書きやすさに関して意見を頂いた．この結果から，ユーザが満足のいくデザインを描けるキャンバスであり，かつレンダリング時間を考慮したキャンバスを採用した．実験に用いたノート PC は Windows 10 Pro, Intel(R) Core™ i5-7200U CPU, RAM 8.00 GB である．

6.1 レンダリング時間

まず，正三角形のキャンバス(正四面体，正八面体，正二十面体)においては，敷き詰められる正三角形の一辺の長さ(a ピクセル)と，キャンバスの正三角形一辺に入る正三角形の数(n 個)に応じて，2 通りのキャンバスのレンダリング時間を比較した(図 15)．ウィンドウサイズに合わせ，キャンバスの正三角形の一辺は 800 ピクセルである．このときに敷き詰められた三角形の総数を N 個とする．図 14 に示した模様を 2 通りのキャンバスで描き，それぞれのキャンバスにおいて 5.2 節で述べた 2 種類のシミュレーションで 5 回ずつレンダリングを行い，そのレンダリング時間を計測した．

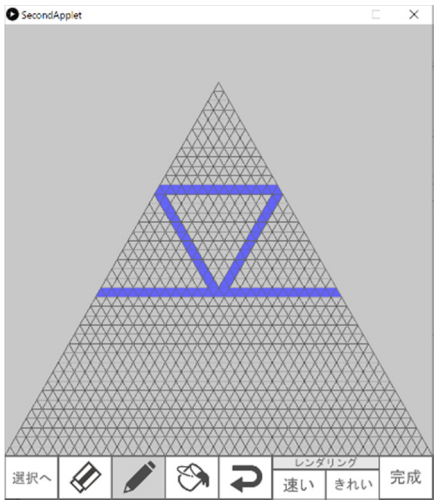
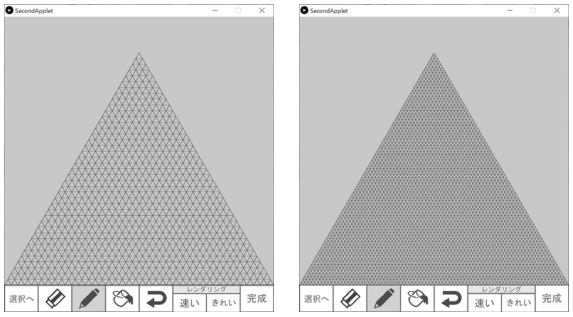


図 14　全キャンバスで共通して描いた模様.



$a=20, n=40$　　$a=10, n=80$

図 15　異なる正三角形のキャンバス.

本システムでのレンダリングでは，レイトレーシング法を用いており，図 16 のようにワールド座標系において視点からスクリーンの画素に向かうレイとユーザが削った画素との交点を計算した．このとき鏡は完全鏡面反射する表面であることから正反射ベクトルをそのまま追跡に使用した．反射回数については 5.2 節で述べた通り，簡易的なシミュレーションでは 1 回，詳細なシミュレーションでは 4 回に設定した．レイと三角形の交差判定には Moller-Trumbore intersection algorithm [15]を用いて，N 個の小三角形すべてに対して，交差判定を行った．交差しないことがわかったらその先の交差判定処理は行わないこととした．計測時間を付録の表 A.1-A.3 に示す．5 回計測した平均値とした．ただし，2 種類のシミュレーションのうち，簡易的なシミュレーションを「速い」，詳細なシミュレーションを「きれい」とした．「速い」のレンダリングを行った場合，三角形の数 n に比例して計算量が増えると考えられる．特に「きれい」のレンダリングを行った場合，2 倍弱の差が出ることがわかった． 高速化の議論については 9 章で考察する．

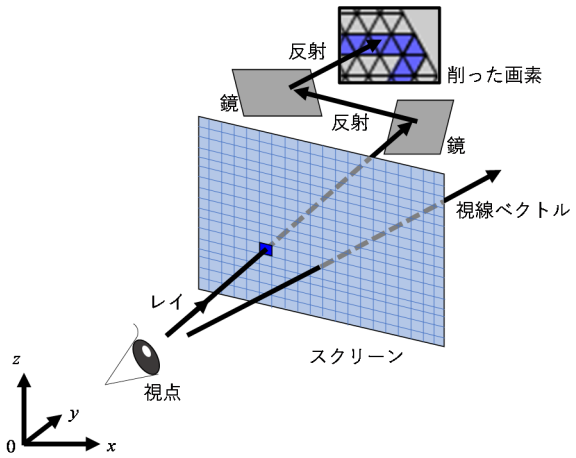
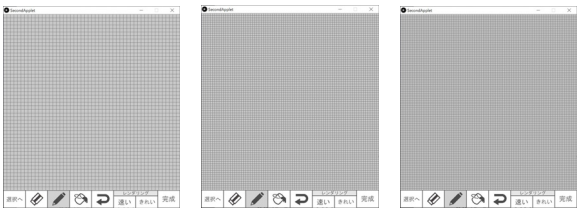
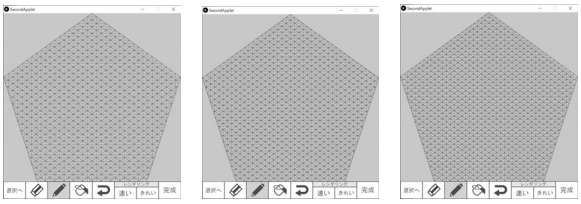


図 16　レイトレーシングの模式図



$n=50$　　$n=80$　　$n=100$

図 17　異なる正方形のキャンバス.



$n=50$　　$n=55$　　$n=60$

図 18　異なる正五角形のキャンバス.

次に，正方形のキャンバス(正六面体)では，キャンバスの正方形一辺に入る正方形の数(n)に応じて，図 14 に示した模様を 3 通りのキャンバスで比較した(図 17)．ウィンドウサイズに合わせ，キャンバスの正方形の一辺は 800 ピクセルである．レンダリング時間の計測結果を付録の表 A.4 に示す．5 回計測した平均値とした．正四面体と同じく，「きれい」のレンダリング時間に大きな差が出ることがわかった．　最後に，正五角形のキャンバス(正十二面体)では，キャンバスの縦向きに敷き詰めることができる三角形の最大個数(n) に応じて，図 14 に示した模様を 3 通りのキャンバスで比較した(図 18)．ウィンドウサイズに合わせ，キャンバスの正五角形の一辺は 500 ピクセルである．レンダリング時間の計測結果を付録の表 A.5 に示す．5 回計測した平均値とした．他のキャンバスほど顕著ではないが，五角形も同じく「きれい」のレンダリング時間に差が出た．

6.2 ユーザスタディ

立体万華鏡の制作経験のない 20-25 歳の男女 5 名に，キャンバスに敷き詰める図形の大きさによる書きやすさについて意見を頂いた．

まず，正三角形のキャンバス（正四面体，正八面体，正二十面体）の場合，敷き詰められる正三角形の一辺の長さが 20 ピクセル($a=20$)でも十分に描きたい模様を描けるという意見で一致した．6.1 節より $a=20$ の方が「きれい」のレンダリングを行う場合にレンダリング時間が短いことが分かっているため，本システムでは図 15 の左側($a=20, n=40$)を採用した．

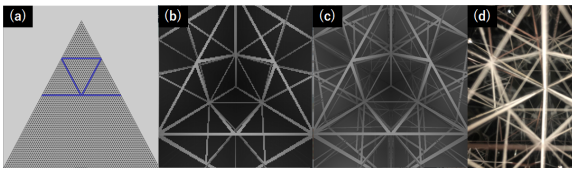
次に正方形のキャンバス（正六面体）の場合，キャンバスの正方形一辺に入る正方形の数が 50 個($n=50$)では十分に詳細な模様を描くことができないという意見が多かったこと，正方形の数が 100 個($n=100$)ではレンダリングに時間がかかりすぎることを踏まえ，正方形の数が 80 個($n=80$)を採用した．

正五面体のキャンバス（正十二面体）の場合，キャンバスの縦向きに敷き詰めることができる三角形の最大個数が 50 個 ($n=50$)でも描けるものの，3 通りの中で最大個数が 60 個($n=60$)の場合が最も模様が描きやすいという意見が多かったため，描きやすさとリアルタイム性のバランスを考え，最大個数が 55 個($n=55$)のキャンバスを採用した．

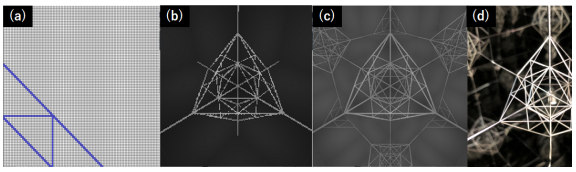
これらの結果から，採用した「速い」のレンダリングを行う場合，レンダリング時間が短くても 1.40 秒かかる．「速い」ボタンを押さずに常時簡易シミュレーション結果を表示することを考慮したが，模様を描画するたびに 1.40 秒かかるシステム設計はユーザにとって快適ではない．そのため，本システムでは「速い」ボタンを押下した際に簡易シミュレーションをすることとした．

7. システムを利用して制作したデザイン結果

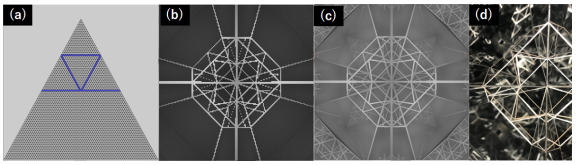
システムを用いてデザインした結果を実際に制作した立体万華鏡の模様と共に図 19 に示す．図 19 において，(a) キャンバス画面，(b) 簡易的なシミュレーション結果，(c) 詳細なシミュレーション結果，(d) 完成した立体万華鏡で視点からみた様子をそれぞれ表している．制作した立体万華鏡の外形は図 20 に示す．システムを用いた模様のシミュレーション結果と，実際に作成した模様の見え方に大きな差は見られなかった．



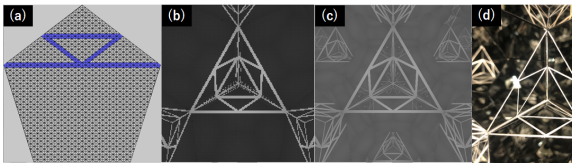
(a) 正四面体のデザイン結果



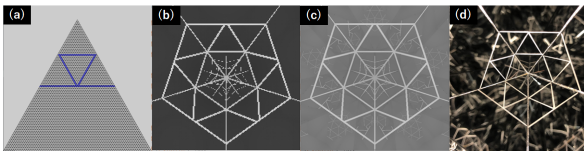
(b) 正六面体のデザイン結果



(c) 正八面体のデザイン結果



(d) 正十二面体のデザイン結果



(e) 正二十面体のデザイン結果

図 19　各正多面体のデザイン結果.

図 20　実際に制作した立体万華鏡.

8. ユーザ実験

本システムの有用性を確認するためにユーザ実験を行った.

表 1　アンケート項目.

	質問項目
問 1	システムへの入力方法を教えて下さい. (マウス, タッチパッド, 指(画面に直接触れる), ペン(液晶画面に使用可能なもの), その他)
問 2	前問で選択した入力方法はデザインしやすかったですか. (1.とてもしづらかった-5.とてもしやすかった)
問 3	システムは簡単に操作できましたか. (1.とても難しかった-5.とても簡単だった)
問 4	システム画面は見やすかったですか. (1.とても見づらかった-5.とても見やすかった)
問 5	デザインの機能で使用したものを全て教えて下さい. (消しゴム, 塗りつぶし, ひとつ戻る, 簡易的なシミュレーション, 詳細なシミュレーション)
問 6	追加で欲しいと思った機能があれば教えて下さい. (自由記述)
問 7	デザインにかかった時間(分)を教えて下さい. (記述形式)
問 8	システム全体を通して, 良かった点があれば教えて下さい. (自由記述)
問 9	システム全体を通して, 改善点があれば教えて下さい. (自由記述)
問 10	満足のいくデザインができたと思いますか. (1.全くそう思わない-5.とてもそう思う)
問 11	差し支えなければ, 完成したデザイン画の提出をお願いします. (画像ファイルのアップロード)

8.1 デザインツール

本システムを用いて初心者が満足のいくデザインができるかについてユーザ実験を行った. 実験参加者は, 20 - 25 歳の男女 9 名であり, 全員が立体万華鏡の制作経験はなかった. また. 実験はオンラインにて実施した. 1) 立体万華鏡について, 2) 立体万華鏡の作り方, 3) システムの使用方法について, を書いた A4 用紙 1 枚を配布した. Processing の実行ファイルを渡して各自の PC で立ち上げて自由にデザインをしてもらい, 事後にアンケートを行った(表 1).

システムの操作性並びに完成したデザインについての満足度と, 使用した機能を尋ねる設問に加えて, 自由記述での, 追加で欲しい機能と. システムの良い点と改善点についての設問を設けた. アンケート結果を図 21 に示す. また. 任意で完成したデザイン画を提出してもらった(図 22).

システムの良かった点としては, 図 21 に示した問 3, 問 4 の結果より, シンプルで見やすく, 直観的に分かりやすいこと, レンダリングが 2 種類あることで, 試行錯誤がしやすいことなどが挙げられ, システムの操作性に関する満足度は高かった.

一方, システムの改善点としては, タッチパッドでのデザインが難しいため, タッチペンやマウスでの操作を推奨した方が良いこと, また, レンダリングが終わったかどうかがわかりにくいため, 終わった時やロード中の表示があると良いというこ

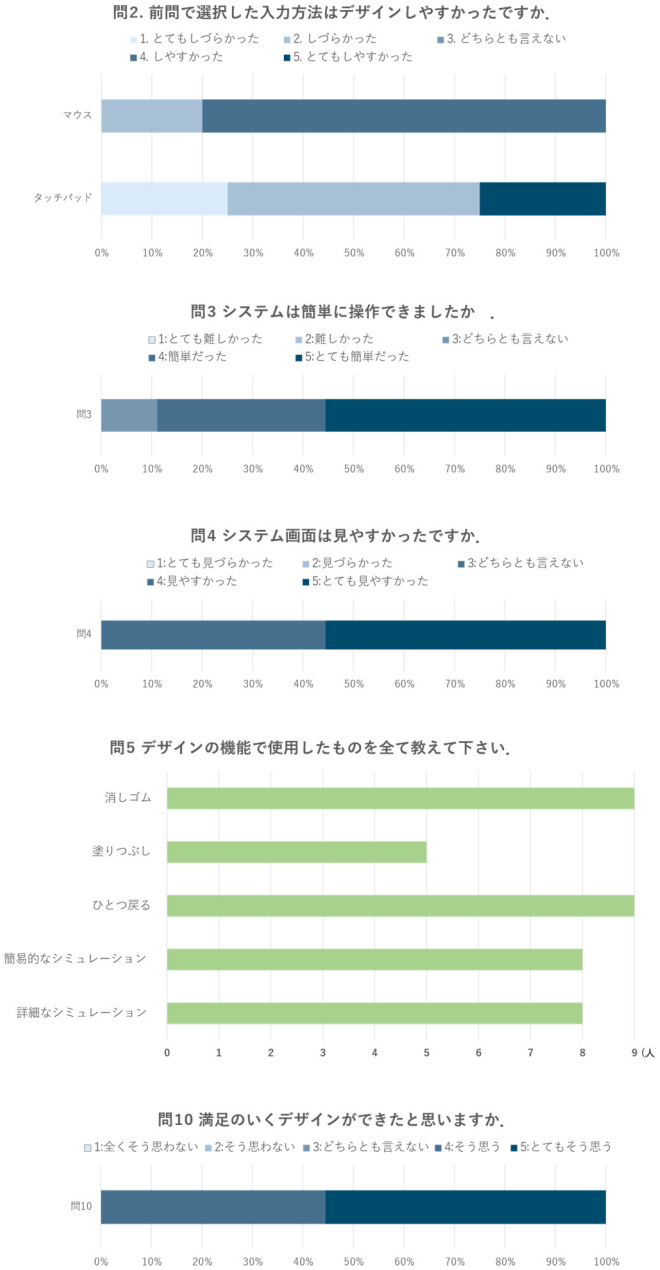


図 21　アンケート結果.

とが挙げられた．実際に，問 1 でマウスが 5 人，タッチパッドが 4 人という結果のうち，マウスでの入力はデザインしやすかったが，タッチパッドでの入力はデザインしづらかったという意見が多くあった．

また，デザインを支援する機能に関しては，どの機能も使われる機会があった．これらの既存の機能を踏まえて，追加で欲しいとされた機能として，画面拡大機能，複数回戻ることができる機能，直線が引ける機能，消しゴムとペンのサイズ変更，ひとつ進む機能(ひとつ戻った後に，やはり戻る前の方が良かったということがあったため)が挙げられた．普段ペイントツールを使用することに慣れていることから，模様をデザインするキャンバスとして，ペイントツールに近しい機能が求められていることがわかった．

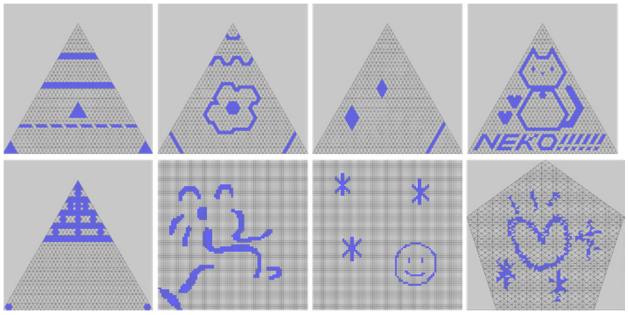


図 22　ユーザ実験での完成したデザイン画.

デザインにかかった時間は 5-20 分であり，問 10 の「満足のいくデザインができたと思うか」という問いに対しては，「4. そう思う」，「5. とてもそう思う」という回答が多く見られたため，デザインの試行錯誤は行えていたようである．

8.2 デザインおよび製作実験

小学生 2 名(ユーザ A: 小 1 女子,ユーザ B: 小 4 男子)に本システムを用いてデザインを行ってもらったあと，実際に制作してもらった(図 23)．まず 8.1 節と同様，事前レクチャーとして立体万華鏡についての A4 用紙 1 枚の紙を配布した．その後，自由にデザインをしてもらった．

システムの最初の画面に表示される注意事項については，ユーザ A は漢字が読めず，ほぼ読まないままデザイン画面へ進んだ．ユーザ B は読んでいたものの既習漢字以外の漢字が多く難しかったようである．デザインではユーザ A は立方体を選択，ユーザ B は四面体を選択していた．選択した理由を聞いたところ，「面がたくさんあるとあとで作るのが大変そうだから」と話していた．どの多面体を選択しても削るのは 3 枚ということを事前に伝える必要があった点は反省点である．

2 人がデザインした結果を図 24 に示す．デザイン中のレンダリングは 2 人とも「速い」を常に使って描画をしていた．ほぼ完成になると「きれい」を選択して確認し，もう少し加えたいときには再度「速い」を使ってデザインしていた．ユーザ A はイチゴのデザインを行い，「速い」でいくつも反射している様子が描かれると歓声を上げて喜んでいた．次に「きれい」を選択した際には，もっとたくさんのイチゴが描画され，重なったりしているのを面白がっていた．ユーザ B は直線的な幾何学図形をベースとしたデザインを描きたがっていた．線を対称に書きたい様子で両サイドからの距離を数えたりしながらポチポチとクリックをするようにデザインをしていた．描画のところでは同じように，歓声を上げながら喜んでいた．

ユーザ A は一度描いたイチゴをもう少し真ん中に動かしたいということができずに悩んでいた．またもう少し大きくしたいといった要望もあった．ユーザ B は最初に何を描きたいか決まらず試行錯誤をしていたこともあってか，最初からやり直したいときに全クリアボタンがほしいとの要望があった．

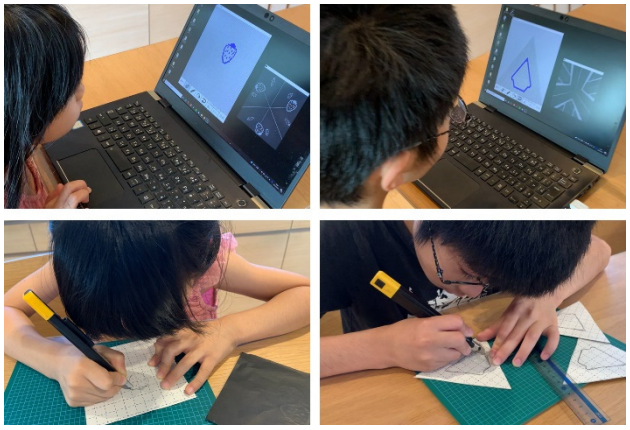
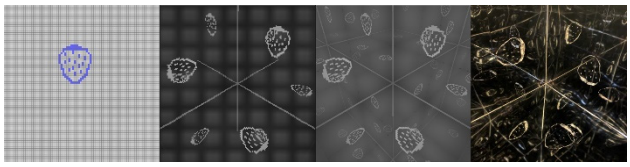
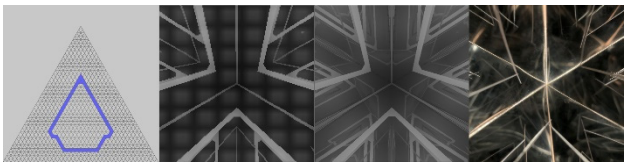


図 23　ユーザ実験の様子.



(a) ユーザ A によるデザイン



(b) ユーザ B によるデザイン

図 24　小学生によるデザインおよび製作.

実際に作る過程では，システムでデザインして保存した画像をプリンタで印刷し，それを型紙としてミラー板を切ったあとカーボン紙で写し取った．カーボン紙を使うことが初めてだったことから 2 人とも喜んでいた．カッターで細かく削っていく作業については 2 人とも特に困らず丁寧に作業をしていた．ユーザ A は細かくカッターを動かし削り取っていた．ユーザ B は直線を組み合わせた図形だったからか定規を使ってカッターで切っていた．削った後は，ユーザ A はイチゴを 3 面どのように貼り合わせたら良いかわからずに悩んでいたことから設計支援に加えて，制作支援があると良いことがわかった．

出来あがり，実際に中を覗いたときには 2 人とも歓声を上げて大喜びしていた．覗いていて喜んだ後，のぞき穴が小さいためもう少し大きく切りたいという要望があった．切ってみたところ，自分の目が写り込んでしまうことに気づき，再度黒テープで貼り直した．のぞき穴の大きさは，今回は経験値で穴を開けたがユーザがデザインツールで大きさを試行錯誤できるようにするのもありかもしれない．デザイン時間は試行錯誤も含めて 10 分程度，制作時間は 30 分程度であった．

9. 改良可能性

本システムで用いたレンダリングの実装方法は，6.1 節に示したように，キャンバス全体の多角形の 1 辺に n 個の小多角形が配置されており，このときの小多角形の総数 N に対して，交差判定を行っている．レイの数を M としたときにその計算量は $O(MN)$ となり，CPU 実装において 1.5 - 2.5 sec の時間がかかっている．今後の改良可能性として Bounding Volume Hierarchy (BVH) などの階層的なデータ構造を用いて高速化を行った場合には，レイと交差判定を行う回数を劇的に減らすことができ，そのときの計算量は $O(M\log(N))$ となる．よって，現在 1.5 - 2.5 sec かかっている簡易描画は 1 ms 程度になりリアルタイム性が増し，ユーザ体験が劇的に改善されると考えられる．BVH を利用する場合には事前に BVH を構築する時間が必要になるが，本システムにおける小多角形の総数 N は高々 10,000 程度であり，実用として問題ないと考える．

また，別の改良可能性としては N 個の小多角形との交差判定を行わず，ユーザがペイントした部分に対してレイが交差するかを判定することで，より単純かつ高速な実装ができると考えられる．分割前の大きな多角形と 1 回だけ交差判定を行い，その三角形上の重心座標からペイント内容を保持したテクスチャの値をルックアップしてペイント部分とレイが交差するかを判定することで実装でき，これにより交差判定の総数が三角形分割数に依存しない実装にすることができ，大幅なシステム高速化が期待できる．

10. まとめと今後の課題

我々は立体万華鏡制作のためのデザイン支援システムを提案した．本システムを用いることにより，制作過程の後戻りができないことに加え，制作に時間がかかるという特徴を持つ立体万華鏡の試行錯誤を容易にすることができる．また，試行錯誤が容易になることで，ユーザの希望に合わせた納得のいく作品を作るための手助けともなり得る．

今後の課題としては，ユーザ実験結果を参考にユーザインタフェースの再検討を行っていく．9 章で述べたように，デザイン段階において要望の高い機能が多くあったため，模様のデザインを行う際のユーザインタフェースの更なる充実を検討する．また，前章で述べたようにレンダリングの実装方法を変更することで高速化が見込めるため今後検討していく．さらに，万華鏡を組み立てる際に，図 25 のように傷を付けることのできる面の一部が不透明なビニールテープでふさがれてしまうことで，傷を付けても光を通さない部分ができてしまうような事情をシステム内にも適応する必要がある．これはキャンバス上で，描画不可能部分とするといった対策を予定している．

また，このようにデザイン支援に重きを置いてきたが，立体万華鏡の制作経験がない人向けの制作支援があると良いこともわかった．例えば，立体万華鏡の展開図を提示することで，模様を付けた面の貼り合わせる向きに迷うことがなくなるような

制作段階を支援する提示も検討する．

今回は正多面体の立体万華鏡のデザインに取り組んだ．正多面体を扱った理由として，空間充填するものとしないものがあること，面の形状がすべて同じであり制作しやすいこと，キャンバスに同じデザインをしても，面の数によって完成した万華鏡の見え方が違うことなどがあげられる．今後，切頂八面体やひし形十二面体といった空間充填の立体や，ユーザが自由に想像した立体で制作できるようなシステムにしていきたい．

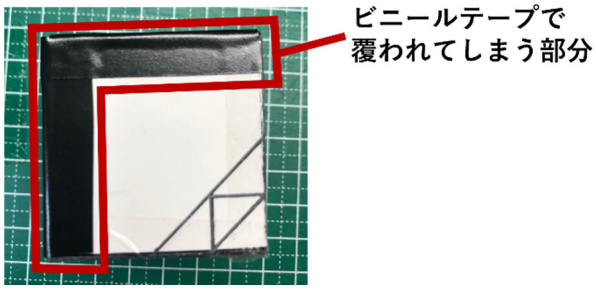


図 25　模様を描くことができない部分．

参考文献

[1] 山見浩司．万華鏡大全．誠文堂新光社，2016.

[2] Brewster, David. The Kaleidoscope: Its History, Theory, and Construction with its Application to the Fine and Useful Arts (2 ed.). J. Murray, 1858.

[3] CUMOS. http://cumos.jp/index.html (2022/01/25 確認)

[4] 本間梨々花，越後宏紀，五十嵐悠紀．RePoKaScope:立体万華鏡のためのデザイン支援システム NICOGRAPH2021, pp.1-8, 2021.

[5] 万華鏡タイル (KaleidoTIle) . http://www.geometrygames.org/KaleidoTile/index.html (2022/01/25 確認)

[6] ibisPaint X(アイビスペイント) . https://apps.apple.com/jp/app/ibis-paint-x/id450722833?ign-mpt=uo%3D8&l=en , (2022/01/25 確認)

[7] 万華鏡カメラ『PrismScope』https://apps.apple.com/jp/app/wan-hua-jingkamera-prismscope/id513652520?mt=8, (2022/01/25 確認)

[8] 東藤絵美，馬場哲晃，串山久美子．Webカメラを利用した万華鏡映像ツールの提案．情報処理学会シンポジウム論文集(インタラクション2011)，pp. 727-728, 2011.

[9] 望月茂徳. デジタル万華鏡. 日本ソフトウェア科学会第22回大会講演論文集, pp. 1-5, 2005.

[10] 五十嵐 悠紀, 五十嵐 健夫，鈴木 宏正. あみぐるみのための3次元モデリングと製作支援インタフェース. 日本ソフトウェア科学会論文誌 『コンピュータソフトウェア』, Vol.26, No.1, pp. 51-58, 2009.

[11] Yuki Igarashi, Takeo Igarashi and Jun Mitani. Beady: Interactive Beadwork Design and Construction. ACM Transactions on Graphics 31(4), Article No. 49, 2012.

[12] 伊藤謙祐，五十嵐悠紀．木目込み細工デザイン支援システム．画像電子学会論文誌, Vol.49, No.4, pp.315-325, 2020.

[13] 越後宏紀, 宮下芳明. PomPom: 紙巻きオルゴール漫画の制作支援システムの提案. 映像情報メディア学会技術報告, Vol.40, No.11, pp.269-272, 2016.

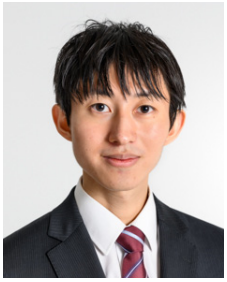
[14] 東芝未来科学館体験イベント(ラビリンスボックス). https://toshiba-mirai-kagakukan.jp/event/event/detail/1260 , (2022/01/25 確認).

[15] Tomas Möller, Ben Trumbore. 1997. Fast, Minimum Storage Ray/Triangle Intersection. Journal of Graphics Tools (JGT) Vol. 2, Issue 1, 21-28.

本間 梨々花

2018年，明治大学総合数理学部先端メディアサイエンス学科入学．コンピュータグラフィックスおよびユーザインタフェースに関する研究に従事．

越後 宏紀



1995 生．2018年明治大学総合数理学部先端メディアサイエンス学科卒業．2020年同大学大学院先端数理科学研究科先端メディアサイエンス専攻博士前期課程修了．同年同大学大学院同研究科同専攻博士後期課程に進学，現在に至る．主にヒューマンインタフェース，教育支援の研究に従事．

五十嵐 悠紀

2010年東京大学大学院工学系研究科博士課程修了．博士（工

学）．日本学術振興会特別研究員DC2，PD，RPDを経て2015年明治大学総合数理学部先端メディアサイエンス学科専任講師，2018年より同准教授．2022年よりお茶の水女子大学准教授，現在に至る．主にコンピュータグラフィックス，ユーザインタフェースの研究に従事． ACM, 情報処理学会， 芸術科学会，他会員.

付録 A レンダリング時間の計測

表 A.1　正四面体のレンダリング時間 (sec).
5 回計測した平均値.

	$a=20, n=40$	$a=10, n=80$
速い	1.40	2.02
きれい	28.38	52.91

表 A.2　正八面体のレンダリング時間 (sec).
5 回計測した平均値.

	$a=20, n=40$	$a=10, n=80$
速い	1.66	2.32
きれい	38.22	67.41

表 A.3　正二十面体のレンダリング時間 (sec).
5 回計測した平均値.

	$a=20, n=40$	$a=10, n=80$
速い	1.82	2.56
きれい	46.15	82.54

表 A.4　正六面体のレンダリング時間 (sec).
5 回計測した平均値.

	$n=50$	$n=80$	$n=100$
速い	1.50	1.95	2.59
きれい	49.00	71.45	102.49

表 A.5　正十二面体のレンダリング時間 (sec).
5 回計測した平均値.

	$n=50$	$n=55$	$n=60$
速い	2.14	2.30	2.46
きれい	56.20	65.14	73.73