Vol.2013-CG-153 No.8
Vol.2013-CVIM-189 No.8
2013/11/28

制約条件と安定性を考慮したレゴブロックによる近似形状構築

北川佑樹[†1]　高井昌彰[†2]　高井那美[†3]

レゴブロックを用いた大規模なアート作品の制作が注目されている．しかし実際の作業時には大量のブロックと広い制作場所を要するため，使用するブロックの種類，配色，個数，組み合わせ方法等を事前に十分検討する必要がある．そこで本研究では，3D ポリゴンモデルデータと使用ブロックの種類や個数の制約条件を入力として与え，実際に作品を配置する場合の物理的安定性や実現の容易さを考慮した上で，入力形状をレゴブロックによって近似表現した 3D モデルを仮想空間に自動構築し，その組み立て配置手順を生成するレゴブロック制作支援システムを開発する．2,200 個のレゴブロックで構築したスタンフォードバニーなど，幾つかの 3D ポリゴンモデルのレゴブロック近似形状構築と実際の作品制作を行い，本システムの有用性と課題点を議論する．

A LEGO Block Builder with Shape Approximation Considering Constraints and Stability

YUKI KITAGAWA[†1]　YOSHIAKI TAKAI[†2]　NAMI TAKAI[†3]

In recent years, a large scale work of art by using LEGO blocks is getting popular. But, beforehand we have to think over what types and how many blocks we need and how we can combine them, because we need so many blocks and a large working space to complete an entire object. In this paper, we propose a LEGO block modeling system which can automatically generate an approximated shape by LEGO block from an arbitrary 3D polygon model considering the constraints such as block resolution. Our system also provides a physical stability evaluation function. We demonstrate actual work production such as a *Stanford Bunny* constructed from more than 2,200 LEGO blocks, and we discuss usability and effectiveness of our system.

1. はじめに

ユニット折り紙や缶アートなど，基本的に同一形状のプリミティブを立体的に多数組み合わせて，任意の 3 次元近似形状を生成する研究が行われている[1,2]．プリミティブの組み合わせ方法の多様性から，同じ最終形を目標として構築しても制作手順，完成形状は制作者によって異なる．また拡張現実（AR）を利用し形状構築の支援を行う研究[3]，教育・知育に利用した研究[4]など，実世界と関連付けられた 3 次元形状構築に関する研究が現在幅広い分野から注目を集めている．

形状構築の玩具の一つとしてレゴブロックがある．レゴブロックはブロック同士を組み合わせて立体形状を形成するものであり，一般的に子供の玩具として世界中で親しまれている．近年では，芸術作品の一種としてレゴブロックによるオブジェが数多く制作されている(図 1)．しかし実際に大規模な制作を行う場合，大量のブロックと広い制作場所を要するため，使用するブロックの種類やそれらの組み合わせ方法，設置した際の物理的な安定性等を，制作前に十分に検討しておく必要がある．

そこで本研究では，3D ポリゴンモデルデータと使用ブロックの種類，個数，完成オブジェの解像度等の制約条件を入力として与え，物理法則に基づいて実際に配置する場合の安定性を考慮した上で，ポリゴンモデルをレゴブロックで近似表現した 3D モデルを自動生成するレゴブロック制作支援システムを開発した．レゴブロックの CAD ツールとして[5]があるが，近似形状の自動構築支援の機能はない．

©レゴ(R)ブロックワールド　SAPPORO
図 1　レゴブロックによるアート作品

2. システムの対象

レゴブロックには様々な形状の基本ブロックがあるが，本システムでは，初心者に最もなじみ深い，1×1，1×2，1×3, 1×4, 1×6, 1×8, 2×2, 2×3, 2×4, 2×6, 2×8 の 11 種類のレゴブロックを扱う．ポリゴンモデルのボクセル化では，

†1 北海道大学大学院情報科学研究科
Graduate School of Information Science and Technology, Hokkaido University
†2 北海道大学情報基盤センター
Information Initiative Center, Hokkaido University
†3 北海道情報大学経営情報学部
Business Administration and Information Science, Hokkaido Information University

Vol.2013-CG-153 No.8
Vol.2013-CVIM-189 No.8
2013/11/28

1×1 ブロックを最小単位とし，ボクセル 1 個に対応させる．

ブロックの配色は，ユーザーの選択に基づき，モデルの色をそのままブロックに反映させるか，あるいはレゴブロックの代表的な 11 色（白，黄，赤，青，緑，黒，黒灰，灰，茶，黄緑，橙）を使用する．これらの 11 色のみを使用する場合には，ブロックに対応するモデルの領域内で使用されている色とのユークリッド距離が最も近い色を選択する．

レゴブロックには一般に推奨されるような「正しい組み方」は存在しない．本システムでは基本的にブロック間の連結性をできるだけ保持するような配置方法をとる．また全体の重量バランスのチェックには，脆弱度の可視化機能を利用でき，組み立て途中の配置方法に重量の影響を適応的に反映させることもユーザーが選択可能である．

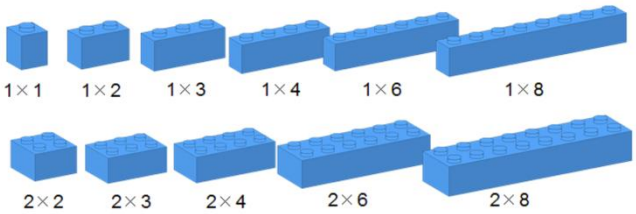


図 2　使用する 11 種類のブロック

3.　システムの概要

本システムの全体の処理の流れを図 3 に示す．本システムはレゴブロックをユーザー自ら 1 ブロックずつ組み上げていくシステムではなく，ユーザーが与えた 3D モデルをレゴブロックのモデルに変換するシステムである．ポリゴンモデルは一般的な 3D モデリングソフトで作成し，実際に組み立てた時のブロック解像度または使用ブロック総個数のいずれかをユーザーが制約条件として指定する．

システムはポリゴンモデルの形状とテクスチャ及び構築の制約条件から，ボクセル化，厚み付け，ブロック配置，連結性判定，再配置処理を行い，最終的にレゴブロックによる 3D モデルを出力する．

また本システムは，生成されたレゴブロックモデルに対してユーザーがインタラクティブに個々のブロックの配置を調整できる GUI も実装している．実際に組み立て可能かどうかを判定するため，荷重とモーメントの計算による脆弱度の可視化の他，ブロック間の結合力を考慮した物理シミュレーションの機能も実装している．

さらに作品の実制作上の支援として，実際の作品展示スペースに設置した際の完成イメージを掴めるよう拡張現実表示機能，ならびに，各層ごとの詳細なブロック配置図を表示する機能を実現している．

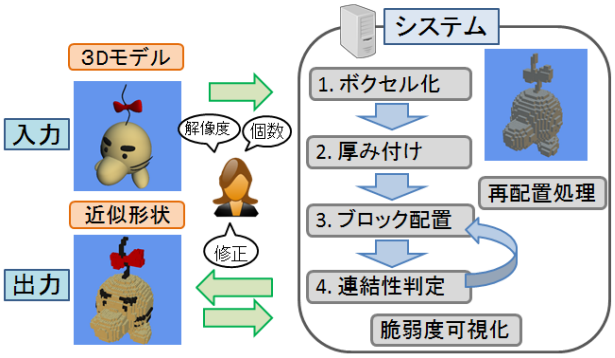


図 3　システムの処理の流れ

3.1 ボクセル化と厚み付け

ボクセル化と厚み付け処理の流れを図 4 に示す．入力された 3D ポリゴンモデルの表面に対して，1×1 ブロックと同サイズ，すなわち縦，横，高さの比が 5 : 5 : 6 の直方体によるボクセル化を行う．

まず内部の厚み付け処理では，層ごとに 1 ブロックずつ走査していき，直下層の同一平面位置にブロックが無く，かつモデルの内部である場合，その部分に 1×1 ブロックを配置する．この処理を最下層から最上層まで繰り返す．

続いて外部の厚み付け処理では，内部の厚み付け同様に，層ごとに 1 ブロックずつ走査していき，直上及び直下の両方の層の同一平面位置にブロックが無く，かつモデルの外部である場合，その層の直下層にブロックを配置する．

また 40 層以上の高さのある構築を行う場合には，内部の厚み付け処理の前に，6 層ごとに内部を全て埋める層を作り，内部を補強する．

これらの内外部の厚み付け処理により，作品全体の強度が増し，非連結ブロックの発生を抑制できる．6 層ごとの内部補強処理は単に強度を上げるだけでなく，実制作時にモデル内の奥にブロックを落とすことを防ぐためでもある．

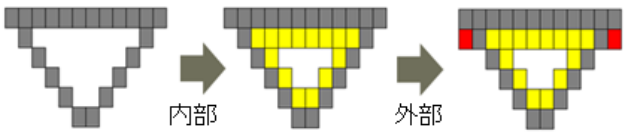


図 4　厚み付け処理

3.2 ブロックの配置

ブロックの配置方法は基本的に 2×8 ブロックから 2×6, 2×4，2×3，2×2 ブロックという順に，2×N ブロックを先に配置し，次に 1×8 ブロックから 1×6, 1×4, 1×2, 1×1 ブロックという順に，1×N ブロックを配置する．どのブロックも一定方向から走査していき，配置可能性を判定する．

2×N ブロックの配置は，はじめに空間を 2×2 ブロック幅

の格子に区切ってから行う．その際，奇数層と偶数層は 1×1 ブロック分だけ縦横に格子をずらし，この格子を基準に配置を行う．1×N ブロックは，奇数層・偶数層ともに 1×1 ブロック幅の格子により配置を行う．図 5 に奇数層と偶数層のブロック配置例を示す．

また 2×2，1×1 ブロック以外の縦横の長さの違うブロックは，奇数層と偶数層でそれぞれ優先して配置させる向きを変更する．

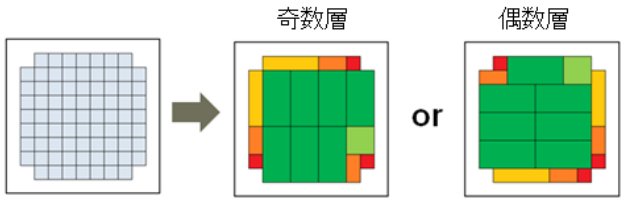


図 5　ブロックの配置

3.3 連結判定

1 ブロックを 1 ノードとし，レゴブロックで構築されたモデルを 1 つのグラフ構造として考える．上下両方に 1 ポッチ（ブロック同士を結合する突起部分）でも接続している場合，その部分を枝とする．したがって 2×8 ブロックの場合は最大で 32 個の枝を持つことになる．このグラフ構造に対して幅優先探索を行い，各ブロックが最下層のブロックのいずれかと連結しているかどうかを調べる．配置予定の全てのブロックの連結性が確認された時，ブロックの配置を決定する．

図 6 で一例を示す．左図のブロック配置をグラフ構造として見たものが右図である．最下層である 1 番のブロックとの連結のない 3, 6 番のブロックを非連結ブロックとして扱う．この非連結ブロックが実際に構築不可能なブロックであり，これらを解消する処理を以降で行う．

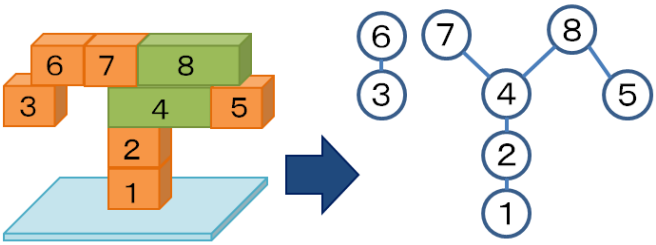


図 6　連結判定

3.4 再配置処理

連結性判定により非連結なブロックが 1 つでも確認された場合，非連結部分の 1 ヵ所を 1×2 ブロックで置き換え，先述したブロック配置ルールに従って，その層のブロックを全て組み直す．次に再配置前と再配置後の総非連結数を比較し，再配置前より再配置後が多い場合または置き換え部分の非連結性が解消されなかった場合，配置を置き換え前に戻す．非連結部分が解消されない場合，1×2 ブロックの置き換え方を変え，組み直しの処理をする．したがって同一箇所で最大 4 回再配置処理を試すことになる．全ての配置でも解消されない場合，置き換える対象のブロックを変えて，組み直しの処理を繰り返す．再配置処理の流れを図 7 に示す．

1×2 ブロックで置き換える理由は，内部・外部の厚み付け並びにブロック配置ルールにより非連結部分が必ず 1×1 ブロックになっているためである．本システムの評価実験で使用したモデルに関しては，再配置処理を繰り返すことで非連結ブロックは全て解消された．

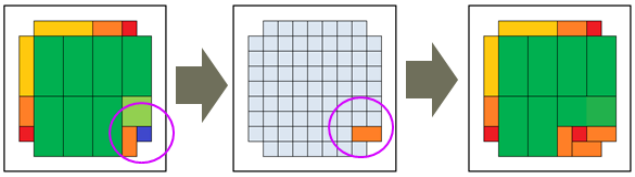


図 7　再配置処理

3.5 脆弱度の可視化

ブロックにかかる荷重とモーメントを計算することによって，水平断面ごとの脆弱度を可視化する．脆弱度を計算する断面の図心を P1，その断面より上側全体の重心を P2，かかる重力を W，上の断面と連結しているポッチ数を N とする．ここで P1P2 ベクトルと W ベクトルによる外積を N で除した値を脆弱度とし，その値を可視化する．また P2 が脆弱度を計算する断面の内部なら，脆弱度はゼロとする．可視化の例を図 8 に示す．同図右で赤色が濃いほど，脆弱度の高い場所であることを表している．

ポッチ部分の摩擦力などによるブロック間の結合力を考慮した物理演算には Physx を利用している．衝突判定を行うプリミティブに円柱型が用意されていないため，直方体を組み合わせたプリミティブで代用している．レゴブロックの材料と同様の摩擦係数を設定することで，現実に近い検証が可能となっている．

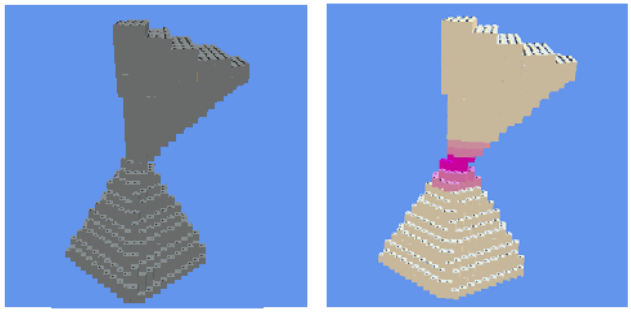


図 8　脆弱度の可視化

Vol.2013-CG-153 No.8
Vol.2013-CVIM-189 No.8
2013/11/28

3.6 脆弱度を考慮したブロック再配置

前節で示した脆弱度が閾値を超えた場合，その部分の値を下げるためのブロック配置処理を行う．モデルの内部において，連結性が保たれる範囲内でブロックを削除し，安定性の増す部分にブロックを配置する．

図 9 で一例を示す．モデル底辺の赤部分の脆弱度が高く，モデル全体が右方向に回転して倒れる恐れがある場合には，同図左の緑部分のブロックを削って，同図右の青部分に再配置し，モデルの重心を左に移動させる．この再配置処理により，モデル底辺部分の脆弱度を下げる．

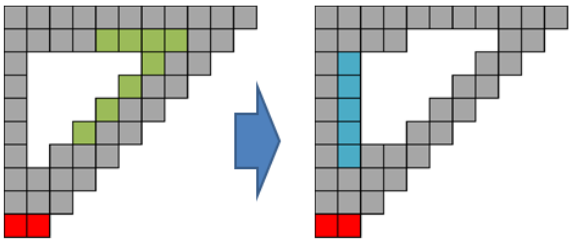


図 9　脆弱度を考慮したブロック配置

3.7 モデル内部の支柱

図 10 に示すような箱型のモデルの天井部分（緑部分）を実際に組み立てる場合，制作者が内部に手を入れて下から支えることができないため，組み立て作業は非常に困難である．この問題を解消するため，内部に一定間隔で 1×2 ブロックの支柱を配置する処理をユーザーが選択できるようにしている．これにより組み立て易さの向上だけでなく，作品全体の強度を高める効果がある．

モデルの内部をブロックですべて埋めつくす方法も考えられるが，全体の総重量の増加や，それに伴う脆弱性の増加，作品の制作コストの観点から，適度に肉抜きされたブロック配置構造が不可欠である．

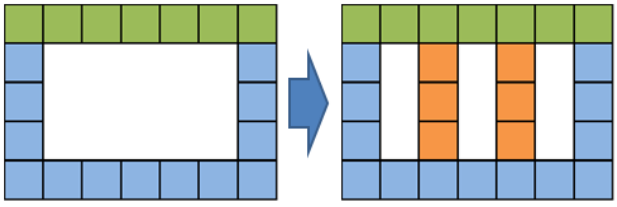


図 10　モデル内部の支柱

3.8 組み立て配置手順の指示

はじめにモデル全体での最下層から，各層に対して閉領域ごとにラベリング処理を行う．次に最下層から走査し，あるラベル発見時に直上層でそれと接続しているラベルの種類を調べる．1 つのラベルと接続している場合，そのラベルと接続する 1 層の他のラベルがあるかを調べる．他のラベルがない場合には，そのラベル領域と統合する．ブロックの配置順は基本的にラベル番号の低い順に決定する．例外として，浮いている部分の配置はその上のラベルの配置後に行う．

図 11 に一例を示す．最下層からラベリングしたのが同図左であり，同図右の黒字の数字が領域統合後のラベル，赤字の数字がブロックの配置順を示す．この処理により，例えば馬などの動物の脚を組むような場合に，全ての脚を層ごと同時並行に組んでいくのではなく，1 本ずつ脚を完成させていくような実際の組み立て工程を考慮した配置順の決定が可能になる．

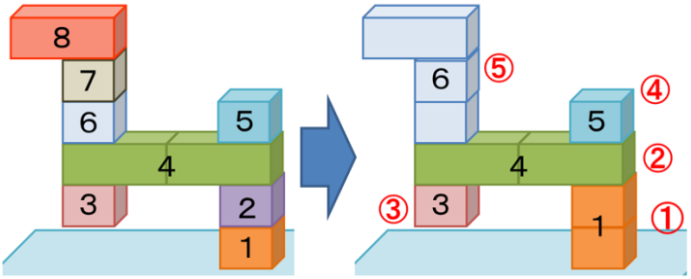


図 11　制作者に対する組み立て配置手順の指示

3.9 手積み機能

図 12 に手積み機能の表示例を示す．右側にモデル全体が表示され，ユーザーはマウス操作により任意の層を選択することができる．選択した部分は赤く示され，左側に選択した層の水平断面が表示される．また，左側の水平断面ではどの種類のブロックが配置されているかの見やすさを考慮し，ブロックのサイズを小さくすることでブロック間に隙間ができるようにしている．ブロックの追加では，使用するブロックの種類をアイコンから選択し，水平断面の任意の個所に合わせることで配置可能である．また任意の個所のブロック削除も可能である．左側の水平断面への操作の結果が右側のモデルにも反映して表示される．

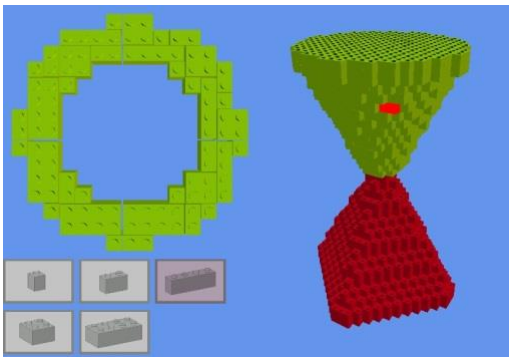


図 12　手積み機能

3.10 拡張現実による表示機能

実際の作品展示スペースに置かれたマーカーの位置に完成オブジェ，各層の断面を拡張現実（AR）で可視化することで実際の制作場所にどれくらいの空間を占めるのか等のシミュレーションが行える．30 層で構築したモデルの AR 表示を図 13 で示す．

図 13　作品の AR 表示機能

3.11 層ごとのブロック配置図

制作者が実際に作品を構築する場合に参考にするものとして，層ごとのブロック配置図の出力機能を用意している．ある層の断面図の様子を図 14 に示す．左が k 層目，右が k+1 層目である．制作者の作りやすさを考慮し，指定層の直下の層の配置についても表示している．また下の層から順に構築していく場合に，1 つ上の層のブロックを配置してからのみ実際に配置が可能になるブロックの表示も同時に行っている．

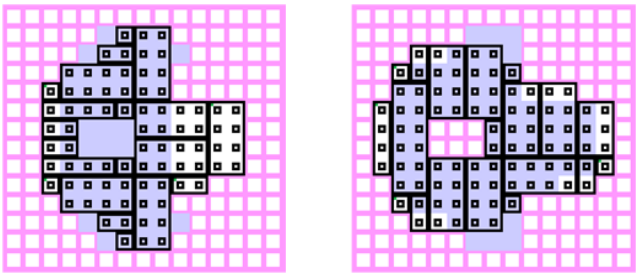


図 14　配置図（k 層(左)と k+1 層(右)）

4　システムの実行結果

4.1 実行環境

本システムの実装においては，開発言語に C#言語，3DCG の描画には XNA4.0，Physx を呼び出すためのラッパーライブラリに StillDesignPhysX.Net，AR 表示には NyARToolkit を使用した．

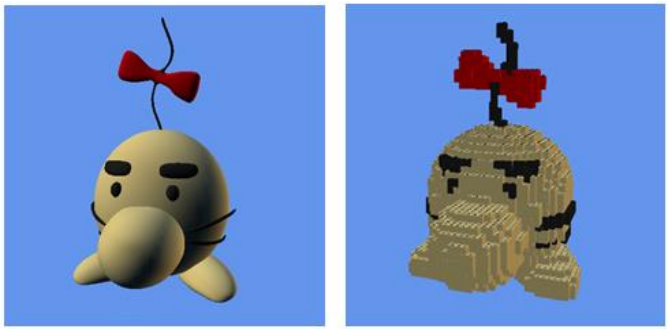
また本システムは，OS: Windows 7 Professional x64, GPU: NVIDIA GeForce GTX 460 の PC 上に実装され，以降の結果も同 PC 上で実行したものである．

4.2 自動構築

本システムを評価するため，いくつかの 3D ポリゴンモデルデータと解像度を本システムに入力し，レゴブロックモデルを生成した．40 層で構築した結果を図 15 に示す．ここでは使用ブロックの個数は制限として与えず，本システムのブロック配置方法において最適な配置結果である．概ね正確にレゴブロックの変換が行われたことがわかる．ブロック間の連結性はすべて満たされているが，図 15(a)のリボンの部分や図 15(b)の耳の部分など，実際に組み立てた場合に崩壊する可能性のある脆弱な部分がある．

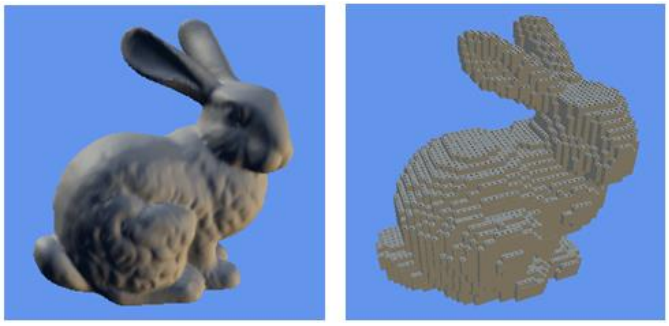
図 15 の 2 つの作品の見積もり情報を図 16 に示す．入力モデルデータの形状により，必要となるブロックの種類が大きく違うことがわかる．構築時間の大半はボクセル化の処理に要する時間である．図 15(a)はポリゴン数 9240，図 15(b)はポリゴン数 34384 であり，同じ解像度でも構築時間の差が出ているのはこのためである．ブロック配置から再配置処理に要する時間はどちらのモデルも 2 秒前後である．また概算費用については，ブロックの 1 ポッチにつき 2 円で計算している．つまり 1×1 ブロックは 2 円で，最大が 2×8 ブロックの 32 円という計算である．

図 15(a)のモデル 1 を異なる解像度で構築した結果を Fig.17 に示す．解像度を上げると入力モデルにより近づくのは明らかであるが，モデル 1 のリボンの部分のように，少ないブロックで上のブロックを支える箇所がある場合には必ずしも解像度を上げるのが良いとは限らない．



©任天堂

(a)モデル 1　(40 層)



(b) モデル 2　(40 層)

図 15　システムの実行結果

Vol.2013-CG-153 No.8
Vol.2013-CVIM-189 No.8
2013/11/28

1×1	306個
1×2	277個
1×3	54個
1×4	106個
1×6	26個
1×8	30個
2×2	52個
2×3	50個
2×4	183個
2×6	51個
2×8	113個
総数	1248個
高さ	38cm
重さ	2220g
概算費用	12,468円
構築時間	10秒

モデル 1

1×1	552個
1×2	500個
1×3	162個
1×4	249個
1×6	52個
1×8	76個
2×2	161個
2×3	87個
2×4	192個
2×6	60個
2×8	112個
総数	2203個
高さ	38cm
重さ	3011g
概算費用	18,336円
構築時間	28秒

モデル 2

図 16　作品の見積もり結果とシステム処理時間

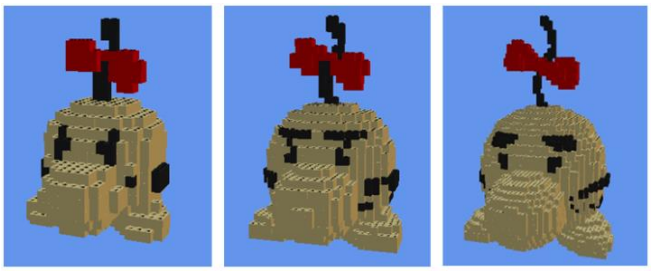


図 17　解像度の違いによる比較 (左から 20, 30, 50 層)

4.3 実制作

本システムがポリゴンモデルから自動生成するブロック配置図をもとに, 図 15 の 2 つのモデルを実際にレゴブロックで構築した結果を図 18 に示す. 作品の完成までに要した制作時間（組み立て作業人数 1 名）は，モデル 1 が 20 時間，モデル 2 が 30 時間であった. モデル 1 のリボン部分やモデル 2 の耳の部分などに崩れやすい脆弱な箇所があるが，ブロックのノリ付けなどの特殊処理は一切せずに全体を構築できている.

図 18　作品の実制作の結果 (モデル 1(左)とモデル 2(右))

5　まとめ

本研究では，仮想空間上にレゴブロックモデルを表現し，与えられたポリゴンモデルの近似形状の自働構築を行い，適切な組み合わせ・配色・脆弱性・総数(コスト)を見積もることにより，作品制作を支援するシステムを実現した. またシステムを使用し実際のオブジェを制作することで，このシステムが有用であることを示した.

今後，よりユーザーの使いやすさを考慮したブロックの配置法や，使用できるブロック種類の拡充，完成モデルの評価方法の拡充を目指す.

参考文献

1) 高橋和茂，高井昌彰，高井那美: “近似形状の構築が可能な缶アート制作支援システム”, 情報処理学会研究報告, Vol.2011-GC-145, No.26, pp.1-6 (2011)
2) 田村友和，高井昌彰，高井那美: “ユニット折り紙を用いた 3 次元メッシュモデルの近似形状構築”, 情報処理学会研究報告, Vol.2010-CG-141, No. 3, pp.1-6 (2010)
3) 川島高志，加藤博一，橘啓八郎: “拡張現実感を用いた 3 次元部品組み立てマニュアルとその評価", 電子情報通信学会技術研究報告, Vol.2001-マルチメディア-101, No.389, pp.1-6 (2001)
4) 後藤顕一，鮫島朋美，高橋三男，松原静郎: “レゴブロックの組み立て再現を利用した表現力育成の基礎的研究: 効果的な記録方法の基礎的なトレーニング", 日本理科教育学会第 59 回全国大会, Vol.日本理科教育学会全国大会要項-202, No.59, pp.1-4 (2009)
5) BlockCAD http://www.blockcad.net/