修 士 論 文

一定枠内に多角形物体を最適配置する手法の開発

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

清水　信行

2002 年 3 月

修 士 論 文

一定枠内に多角形物体を最適配置する手法の開発

指導教官　浅野哲夫　教授

審査委員主査　浅野哲夫　教授
審査委員　中野浩嗣 助教授
審査委員　金子峰雄　教授

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

010053 清水　信行

提出年月: 2002 年 2 月

概 要

最適配置に関する研究は, 一定の生地から洋服の部品を切り取る問題や, VLSI の設計における素子配置問題として様々な分野で行なわれている.

本稿では, 縦横比を固定した枠の中に多角形物体を隙間なくしかも効率よく配置する問題について考察する. さらに宣伝用広告チラシの自動生成に応用できるよう, 見栄えを考慮した配置調整をおこない, より実用的なシステムを構築する

目 次

図 目 次

第1章　はじめに

1.1　研究の背景

最適配置に関する研究は, 一定の生地から洋服の部品を切り取る問題 [3][4][5] や, VLSI の設計における素子配置問題 [1][2] として様々な分野で行われている. 一定の生地から洋服の部品を切り取る問題は, 詰めるのではなく一定の領域から多角形を切り離すという本研究とは逆の性質のものである. これは, 枠の幅が固定され高さが無限である長方形の中で, 多角形の回転を許し配置し生地を切り離す問題である. VLSI の設計における素子配置問題は, 与えられた大きさのチップを対象とし面積を最小にするように長方形の中に配置するというものである. これらの配置問題は, 多項式時間で計算が不可能と思われる非常に困難な問題 (いわゆる NP 困難問題) であるが, 実用的なアルゴリズムが提案され十分な成果が得られている.

これらの分野とは多少性質は異なるが, 一定枠の中に物体を配置していく研究に関して, 商品写真を一定枠内に配置していき宣伝用広告チラシを自動生成するという応用が可能である. 現在, 宣伝用広告チラシ作成に関しては, 専門家 (デザイナー) が独自の視点で商品写真を配置・編集している為, 多大な時間とコストと感性が必要になっているが, 自動配置システムを構築する事によりこれらの大幅な削減が期待できる. しかし, 宣伝用広告チラシのような一定枠内に物体を配置していく研究に関しては, 現段階で十分に研究されていないのが現状である.

1.2　研究の目的

本研究では, 縦横比を固定した一定枠の中に多数の多角形物体を隙間なく, しかも効率よく配置する事を目的する. 多角形の最適配置問題に関しては, すべての配置パターンを考えると NP 困難になる事が知られている. そこで本研究では複数の配置手法を用い, 段階的に物体を配置・調整を行う事により, 計算時間短縮と効率的な最適配置を実現する. さらにこの配置アルゴリズムを宣伝用広告チラシの自動生成に応用し, より実用的なシステムを構築する.

1.3　研究の特色

本研究では, 縦横比を固定した枠を拡大縮小しながら, 任意の凸多角形を隙間なくしかも効率よく配置していく. 凸多角形を配置する上で以下の特色がある.

- **凸多角形を長方形に近似して初期配置**

物体の初期配置に関しては, 凸多角形をいきなり一定枠に配置していくのは大変難しい. そこで本研究では, 物体配置を容易にするため, 凸多角形を長方形に近似して初期配置を行い, その後長方形を凸多角形に復元して微調整するという手法を用いる.

- **ビンパッキング手法を用いた長方形配置**

長方形の初期配置に関しては, ビンパッキングを用いた手法を適用する. 元々ビンパッキングは, 幅の概念だけをもつ１次元パッキングであるが, それに高さの要素を加えて２次元パッキングに拡張する. 代表的なビンパッキング手法として, NextFit 法, FirstFit 法, BestFit 法の３つが知られているが, 過去の実験結果や論文等を考慮して BestFit 法を用いる. また, ビンパッキングはビンの最大長を指定しないと実行できないので, ２分探索法に基づく方法でビンの長さを決定する.

- **３つの配置パターン**

長方形の配置順に関しては, 後ほど宣伝用広告チラシに応用する事を考慮して, 高さを基準に昇順, 降順にソートしたり, ランダムに並べかえたりして３パターン行なう.

- **長方形間のコンパクション**

長方形間の空きスペースをできる限り少なくし占有率を高くする為, 長方形間のコンパクションを実施する. コンパクションは, VLSI の素子配置の為に開発された BSG 構造に基づく方法を適用して行なう. また BSG コンパクション適用後空きスペースがある場合占有率を高められる可能性があるため, 各多角形を縦または横方向に交互につめていく.

さらに, 宣伝用広告チラシに応用するためには見栄えを良くする必要がある. そこで近似長方形を凸多角形に復元した後, 多角形間の空きスペースが均一になるよう配置調整を実施する. 配置調整に関しては以下の特色がある.

- **デローネイ三角形分割の概念を用いた空き領域発見**

配置調整を行なうには多角形の回りにどのくらいの領域が存在するか知る必要がある. その方法としてデローネイ三角形分割の概念を利用して空き領域を見出す.

- **多角形を移動するための方法として以下に述べる 2 つの方法を考える**

１つ目は, 多角形を取り囲む領域の最大内接円を求め, その内接円の中心に多角形を移動させる方法である. 具体的には, 各多角形の包含円と, 各多角形を取り囲む領域の最大内接円を求め, ２つの円の距離が最も大きい多角形を移動させる. この作業を, ある程度収束するまで繰り返し終了する.

２つ目は, 多角形と空き領域とのミンコフスキー和を計算して移動させる方法である.

具体的には, 各多角形の頂点を取り囲む最大領域を見つけ, その領域と各多角形のミンコフスキー和を計算し, その和が最大である多角形を移動させる. この作業も, 何度か繰り返し終了する. ミンコフスキー和の概念を用いる利点としては, 大きさの類似した多角形が一箇所に集中するのを軽減する事があげられる.

1.4 本論文の構成

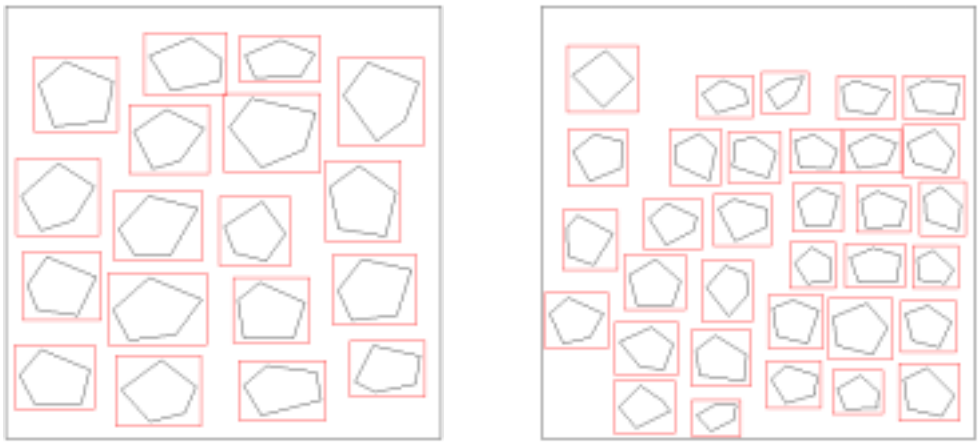
本論分の構成は以下の通りである.
第２章では, ビンパッキング手法を用いて多角形の初期配置をおこなう.
第３章では, 多角形間の隙間を埋めるためＢＳＧ構造を用いてコンパクションを行なう.
第４章では, 本研究の配置アルゴリズムを宣伝用広告チラシに応用するため, 多角形間の配置調整を行なう.
第５章では, 本研究の実験結果をふまえた考察を行なう.

第2章　多角形の初期配置

本章では, 多角形の初期配置について述べる. 対象となる物体は凸多角形であるが, 凸多角形をいきなり配置していくのは大変難しい. そこで本研究では物体配置を容易にするため凸多角形を長方形に近似し初期配置を行なう. 長方形の初期配置に関しては, ビンパッキングを用いた手法を適用する. この手法を用いると大まかではあるが長方形の配置が可能となる.

2.1　凸多角形の長方形近似

与えられた凸多角形に対する包含長方形を求める. 包含長方形は, それぞれ凸多角形のx,y座標に対する最大値と最小値, x_{max},x_{min},y_{max},y_{min} を求め, 2点 (x_{min},y_{min}) と (x_{max},y_{max}) を対角の頂点とする長方形とする. ただし後述の処理において多角形同士が接触するのを避けるため, 予め一定の余裕幅dをもたせる. 従って長方形の包含長方形は2点 $(x_{min}-d, y_{min}-d), (x_{max}+d, y_{max}+d)$ を対角とする長方形となる. 図2.1に, 対象となる凸多角形とその近似長方形を示す.



(a) 多角形18個　　(b) 多角形33個

図 2.1: 対象物体の長方形近似

2.2　ビンパッキングを用いた長方形パッキング手法

長方形パッキングでは, 一定枠に対する長方形の占有率をできる限り高くする事を目的としている. この節ではビンパッキング手法について検討し, 占有率向上を目指した長方形配置を行なう.

2.2.1　ビンパッキング手法について

ビンパッキング手法は, 与えられた１方向のデータをビンに詰めていき, 最後のデータが入力された時ビンの数を最小にするものである. 代表的なビンパッキング手法として, NextFit 法, FirstFit 法, BestFit 法の３つが知られている. この３つの手法を例を用いて紹介する. まず最初にビンの長さは 100 とし, 入力値としてデータ I を与える.

$$I = (55, 23, 28, 45, 42, 5, 20, 71, 33, 75, 63, 34, 5, 14, 13, 28)$$

NextFit 法

NextFit 法は, 与えられたデータを与えられた順番に, ふたの開いているビンに入るかどうか調べ, ビンに入るときは入れ, 入らないときはそのビンにふたをして新しいビンを用意しふたを開けて入れる手法である. 図 2.2 に入力値 I を与えたときの結果を示す. 図 2.2 よりビンの数は 8 になる.

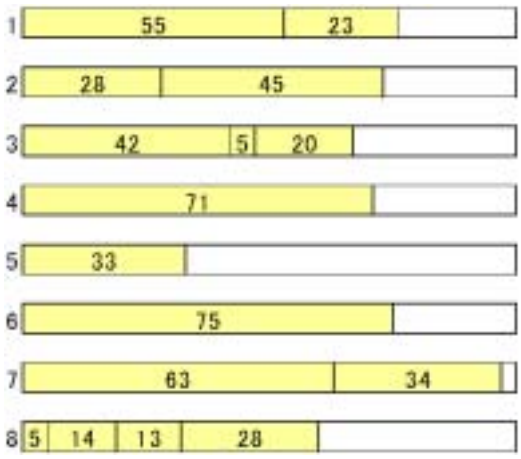


図 2.2: NextFit 法

FirstFit 法

FirstFit 法は, 与えられたデータを与えられた順番に, ふたの開いているビンにふたを開けた順番に入るかどうか調べ, 最初に入るビンに入れる方法である. ただしふたの開いたどのビンにもデータが入らないときは, 新しいビンを用意しふたを開けて入れる. この手法は, NextFit 法と異なりビンにふたをしない. 図 2.3 に入力値 I を与えたときの結果を示す. 図 2.3 よりビンの数は 7 になる.

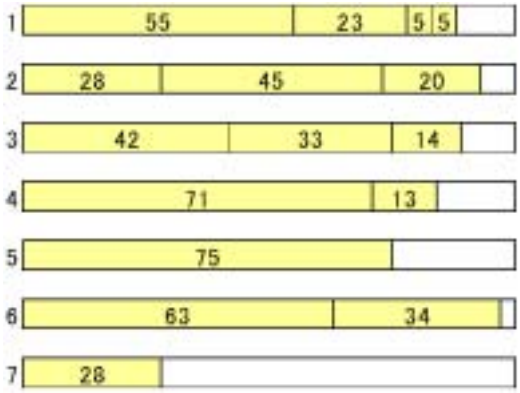


図 2.3: FirstFit 法

BestFit 法

BestFit 法は, FirstFit 法のように最初に入るビンにデータを入れるのではなく, ビンの残りスペースが最小になるビンを選択して入れる手法である. この手法も FirstFit 法と同じくビンにふたをしない. 図 2.4 に入力値 I を与えたときの結果を示す. 図 2.4 よりビンの数は 6 になる.

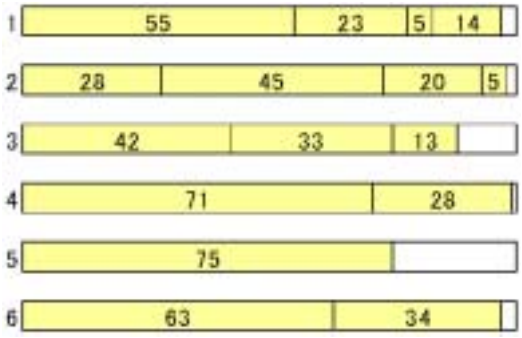


図 2.4: BestFit 法

以上３つの手法でデータ配置を行なうと, ビンの数は, NextFit 法=8, FirstFit 法=7, BestFit 法=6 となり, BestFit 法を用いた時に最も良い結果が得られる. また BestFit 法は, 異なった入力値を与えた場合でも良い結果が得られる傾向がある. この例題では, 与えられた順番に入力値を配置していったが, 入力値を降順または昇順にソートしてやればさらによい結果が得られる可能性がある.

2.2.2 ２次元への拡張

ここまで述べてきたビンパッキング手法は, １方向のデータのみを扱う１次元パッキング手法であった. しかし長方形の配置を行なう場合, 高さと幅の概念を持つ２次元の概念が必要となる. そこで１方向の長さのみ考慮していたビンに対し, 高さの概念を加え２次元パッキング手法に拡張する.

長方形の配置順は, 後ほど宣伝用広告チラシに応用する事を考慮して, 高さを基準に昇順, 降順ソートしたり, ランダムに並べかえたりして 3 パターン行なう. ランダムついては, サンプルデータを 15 個用意し, 配置後最も占有率が高くなるものを採用する. また, ビンパッキングはビンの最大長を指定しないと実行できないので, ２分探索法に基づく方法によりビンの長さを決定する.

2.2.3 実験結果

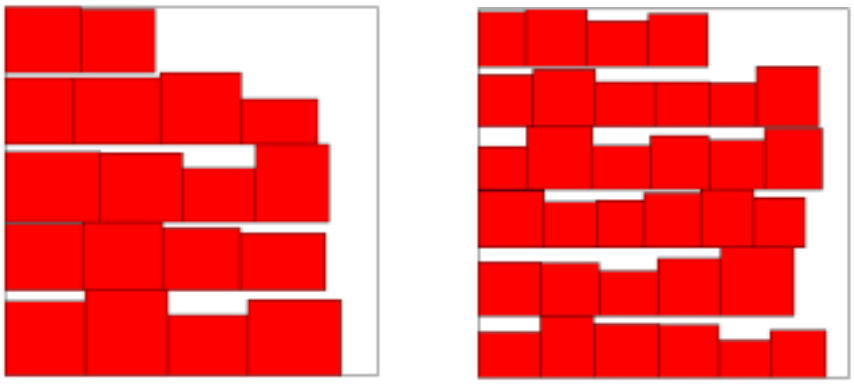
2 次元ビンパッキング手法を用いて長方形の初期配置を実施した. その結果を図 2.5 (ランダム順), 図 2.6 (降順ソート順), そして図 2.7 (昇順ソート順) に示す. また, それぞれの配置に対する占有率を表 2.1 示す.

表 2.1: 一定枠に対する多角形の占有率

	多角形 18 個	多角形 33 個
ランダム順	0.702648	0.72867
降順ソート順	0.822328	0.84566
昇順ソート順	0.749118	0.84566

ランダム順入力

占有率=0.702648　　占有率=0.728667



(a) 多角形 18 個　　(b) 多角形 33 個

図 2.5: 2 次元パッキング結果 (ランダム順)

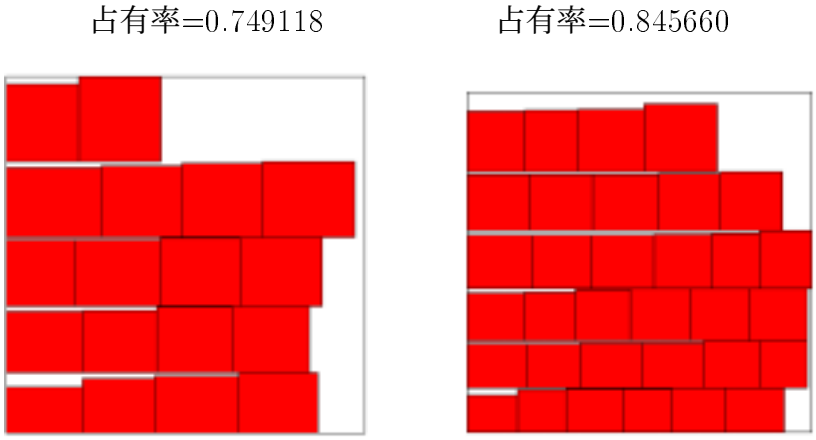
降順ソート順入力

占有率=0.822328　　占有率=0.845660

(a) 多角形 18 個　　(b) 多角形 33 個

図 2.6: 2 次元パッキング結果 (降順ソート順)

昇順ソート順入力



(a) 多角形 18 個　　(b) 多角形 33 個

図 2.7: 2 次元パッキング結果 (昇順ソート順)

2.2.4 評価

占有率は, 多角形を降順または昇順に配置していくと, ランダム順に配置していくより高くなる傾向がある (表 2.1 参照). しかし多角形をソートして配置すると, 枠内の上下に大小の多角形が密集してしまう傾向がある (図 2.6, 図 2.7 参照). だがランダム順に配置していくと, 占有率は低くなるものの大小の多角形がばらばらに配置される傾向にあり (図 2.5 参照), 後ほど宣伝用広告チラシに応用する事を考慮すると, ランダム順に配置していくのが良いものと思われる.

第3章　2次元コンパクション

前章の2次元ビンパッキング手法では長方形の高さを基準にビンを作成するため, 高さ方向に空きスペースが発生する可能性がある. 宣伝用広告チラシに応用する事を考えると, できる限り多角形の占有率を高くし目立つようにしたい. 本章では2次元コンパクションを実施し, 占有率を高める事を試みる.

3.1　BSG構造を用いた長方形コンパクション

長方形間のコンパクションを行なう手段として, VLSIの素子配置の為に開発されたBSG(Bounded-Sliceline Grid)構造[6][7]に基づく方法を適用する.

3.1.1　BSG構造について

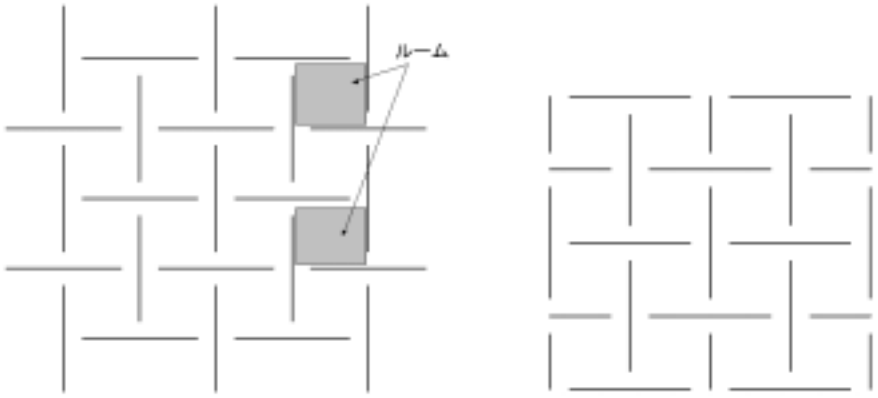
BSGとは, BSG-unitと呼ばれる2単位長の垂直及び水平線分により構成されるグリッドである. 同じ方向のunitは, 1単位ずつずらし重なりなく配置する. Unit集合U_{BSG}は, 式(3.1)のように定義され, 水平線分$H_{i,j}$(式(3.2)), 垂直線分$V_{i,j}$(式(3.3))により構成される. 水平線分$H_{i,j}$と垂直線分$V_{i,j}$を図3.1(a)示す. 式(3.2), 式(3.3), そして図3.1(a)より, 水平Unitは$j-1<y<j+1$に可動し, 垂直Unitは$i-1<x<i+1$に可動する事がわかる.

また４つのBSG-unitにより囲まれた領域をルームと呼ぶ. BSGは全平面で定義される無限位相構造であるが, 長方形のコンパクションに適用する時はその有限部分に着目する. 図3.1(b)に, $4*4$のルームを持つBSGを示す.

$$U_{BSG} = \{V_{i,j} \mid i+j : even\} \cup \{H_{i,j} \mid i+j : odd\}, \tag{3.1}$$

$$H_{i,j} = \{(x,y) \mid i-1 < x < i+1, y = j\}, \tag{3.2}$$

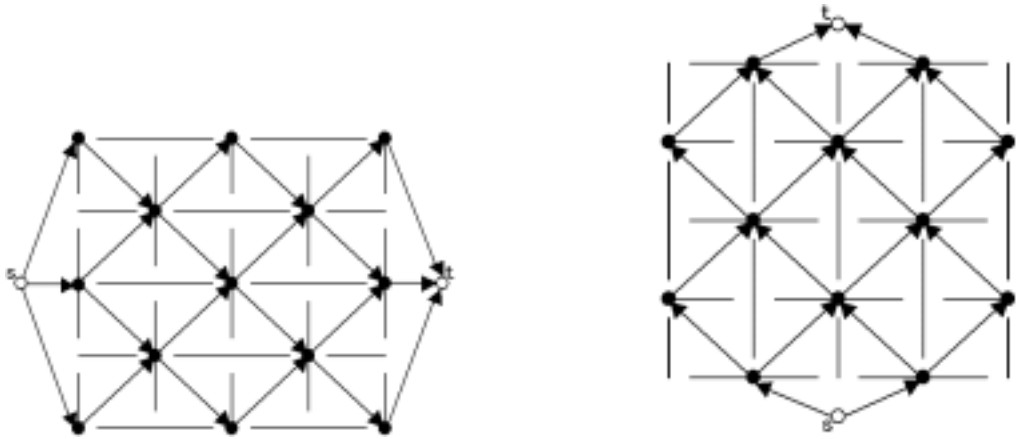
$$V_{i,j} = \{(x,y) \mid x = i, j-1 < y < j+1\}. \tag{3.3}$$



(a) BSG　　(b) $4 * 4$ BSG

図 3.1: Bounded-Sliceline Grid

BSG-unit から, それぞれ水平, 垂直隣接グラフが定義できる. 図 3.2 に, それぞれの隣接グラフを示す. それぞれの隣接グラフより, ソース s からシンク t までの最大パスを, 幅, 高さとする.



(a) 垂直隣接グラフ　　(b) 水平隣接グラフ

図 3.2: unit 隣接グラフ

3.1.2　コンパクションへの適用

BSG を 2 次元コンパクションに応用する.
以下に, 例を用いてコンパクションの手順を示す

Step1 (BSG への配置)
ビンパッキングで配置された長方形を, BSG に配置する. 図 3.3 は, ビンパッキングで配置された長方形を, BSG に配置したものである.

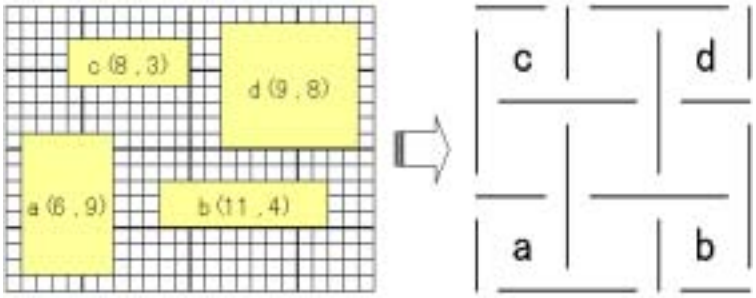
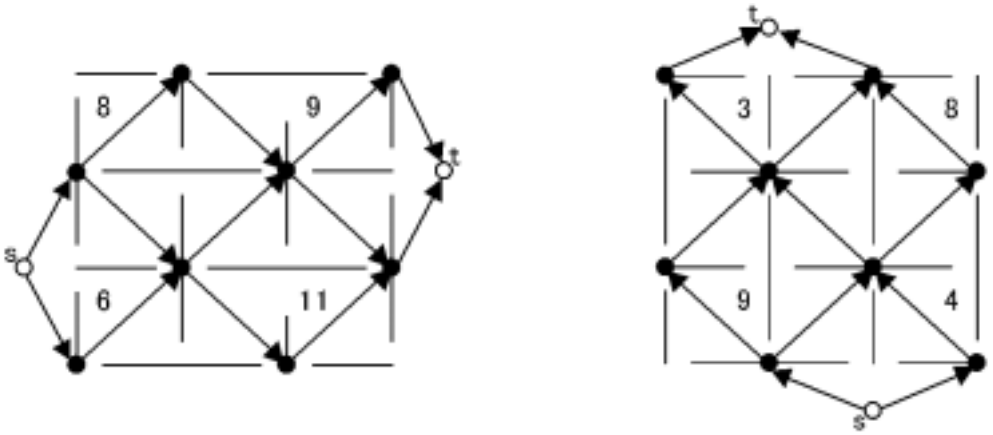


図 3.3: BSG への配置

Step2 (各隣接グラフの最大パスを計算)
垂直隣接グラフと水平隣接グラフの最大パスを計算する. 図 3.4 より垂直隣接グラフの最大パスは 17, 水平隣接グラフの最大パスは 12 となる. 従って幅は 17, 高さは 12 となる.



(a) 垂直隣接グラフ　　(b) 水平隣接グラフ

図 3.4: unit 隣接グラフの最大パスの計算

Step3 (コンパクション完了)

BSG 構造を用いて長方形を再配置した結果は, 図 3.5 のようになる.

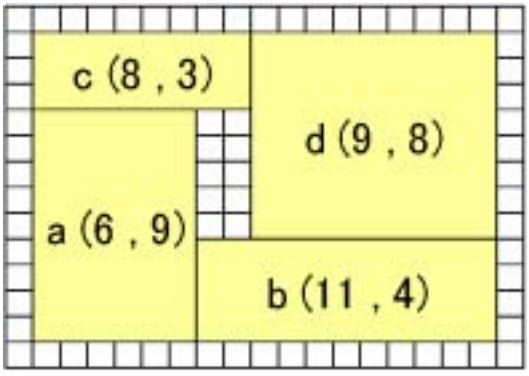


図 3.5: 長方形の再配置結果

3.1.3 実験結果

BSG 構造に基づく長方形間のコンパクションを実施した. BSG への長方形配置に関しては, 経験則に基づき水平方向は 3 ルーム, 垂直方向は 1 ルーム空けた. その結果をそれぞれ図 3.6(ランダム順), 図 3.7(降順ソート順), 図 3.8(昇順ソート順) に示す. また表 3.1 にビンパッキングと BSG コンパクション実施後の占有率と, BSG コンパクションによる占有率の改善度を示す. 表 3.1 より, ランダム順入力に対する占有率は, 多角形 18 個の時 約 2.4 %, 多角形 33 個の時 約 7.0 %改善された. しかし, 降順, 昇順ソート順入力に対する占有率は, 多角形 18 個, 多角形 33 個のどちらも改善はなかった.

表 3.1: 各占有率と BSG コンパクションによる占有率改善度

(a) 多角形 18 個

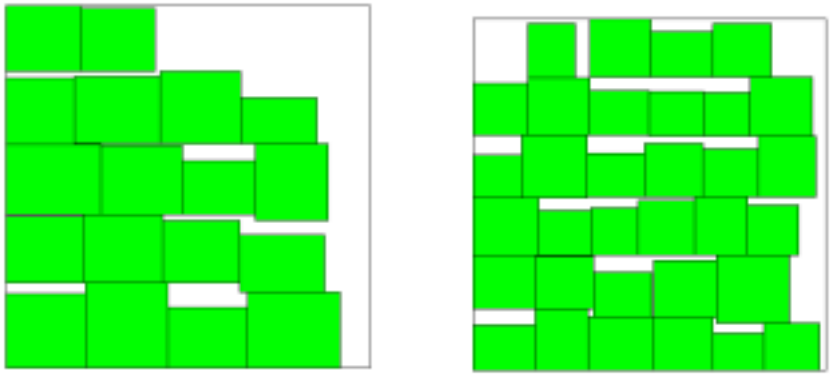
	占有率		
	ビンパッキング	BSG コンパクション	改善度
ランダム順	0.702648	0.726874	0.024226
降順ソート順	0.822328	0.822328	0
昇順ソート順	0.749118	0.749118	0

(b) 多角形 33 個

	占有率		
	ビンパッキング	BSG コンパクション	改善度
ランダム順	0.728667	0.798690	0.070023
降順ソート順	0.845660	0.845660	0
昇順ソート順	0.845660	0.845660	0

ランダム順入力

占有率=0.726874　　占有率=0.798690

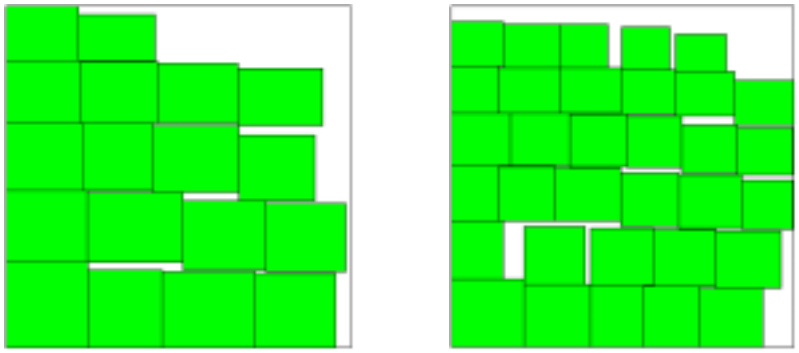


(a) 多角形 18 個　　(b) 多角形 33 個

図 3.6: BSG コンパクション結果 (ランダム順)

降順ソート順入力

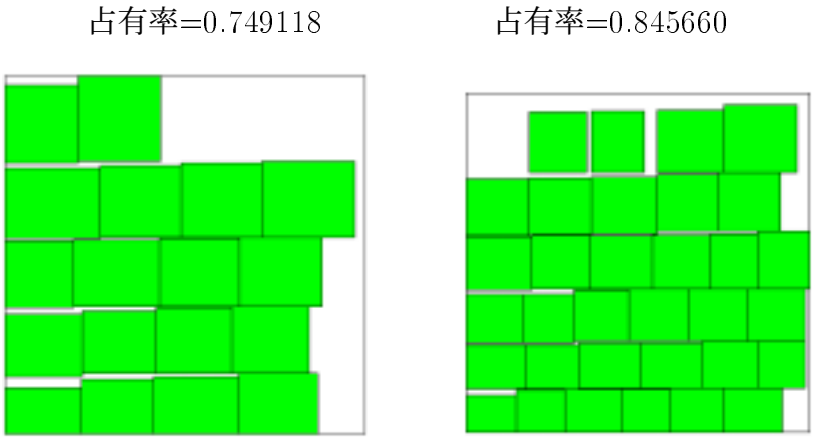
占有率=0.822328　　占有率=0.845660



(a) 多角形 18 個　　(b) 多角形 33 個

図 3.7: BSG コンパクション結果 (降順ソート順)

昇順ソート順入力



(a) 多角形 18 個　　(b) 多角形 33 個

図 3.8: BSG コンパクション結果 (昇順ソート順)

3.2 追加コンパクション

BSG コンパクションを実施後, まだ余分な領域がある場合さらに占有率を高くできる可能性がある. そこで, 各多角形を縦または横方向に交互に詰めてコンパクションを実施する.

3.2.1 実験結果

追加コンパクションを実施したそれぞれの結果を, 図 3.9(ランダム順), 図 3.10(降順ソート順), 図 3.11(昇順ソート順) に示す. また表 3.2 に BSG コンパクションと追加コンパクション実施後の占有率と, 追加コンパクションによる占有率の改善度を示す. 表 3.2 より, ランダム順入力に対する占有率は, 多角形 18 個の時 約 2.9 %, 多角形 33 個の時 約 3.0 %改善された. 降順ソートに対する占有率は, 多角形 18 個, 多角形 33 個のどちらも改善されなかった. 昇順ソート順に対する占有率は, 多角形 18 個の時 約 2.0 %改善されたが, 多角形 33 個の時は改善されなかった.

表 3.2: 各占有率と追加コンパクションによる占有率改善度

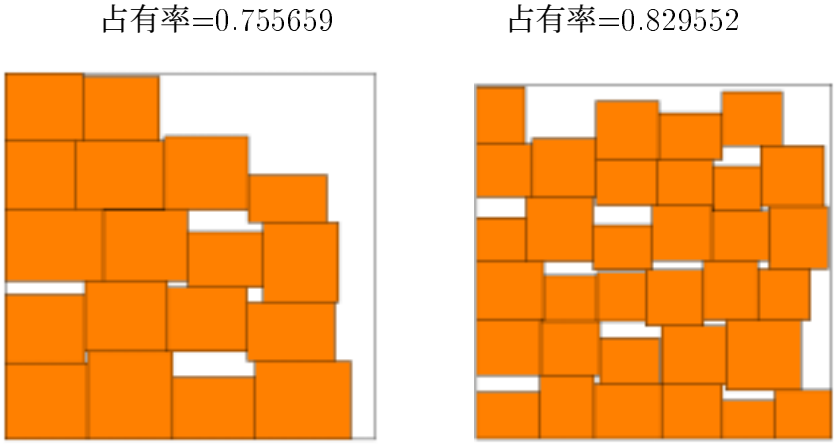
(a) 多角形 18 個

	占有率		
	BSG コンパクション	追加コンパクション	改善度
ランダム順	0.726874	0.755659	0.028785
降順ソート順	0.822328	0.822328	0
昇順ソート順	0.749118	0.769003	0.019885

(b) 多角形 33 個

	占有率		
	BSG コンパクション	追加コンパクション	改善度
ランダム順	0.798690	0.829552	0.030862
降順ソート順	0.845660	0.845660	0
昇順ソート順	0.845660	0.845660	0

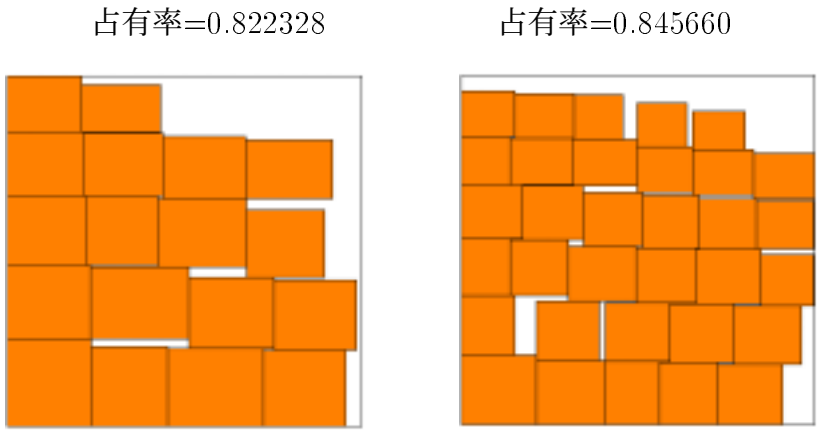
ランダム順入力



(a) 多角形 18 個　　(b) 多角形 33 個

図 3.9: 追加コンパクション結果 (ランダム順)

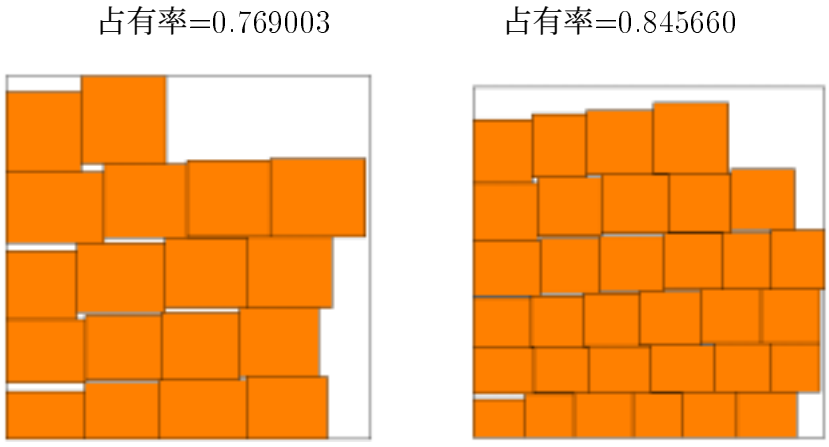
降順ソート順入力



(a) 多角形 18 個　　(b) 多角形 33 個

図 3.10: 追加コンパクション結果 (降順ソート順)

昇順ソート順入力



(a) 多角形 18 個　　(b) 多角形 33 個

図 3.11: 追加コンパクション結果 (昇順ソート順)

3.3 評価

多角形間のコンパクションにより得られた占有率の改善度を表 3.3 に示す.

表 3.3 より, 降順または昇順ソート順に関しては, 初期配置の段階で占有率が高かった事もあり, コンパクションによる改善はなしか, もしくは低かった. だがランダム順に関しては, コンパクション実施により降順や昇順ソート順に近い占有率を得る事ができた. 多角形 18 個の場合は約 5.3 %, 多角形 33 個の場合は約 10.0 %, 占有率を高める事ができた.

宣伝用広告チラシに応用する場合, 多角形をソートして配置していくと, 占有率は高いものの枠内の上下に大小の多角形が密集する傾向がある. 宣伝用広告チラシの用途にもよるが, 人間の美的感覚に合わない可能性がある. しかしランダム順に並べていくと, 占有率は低いものの密集する場合が少ない. 従ってコンパクションを実施する事により, ランダム順に初期配置したとしても, ある程度まで占有率を高められる可能性がある事が分かった.

表 3.3: コンパクションによる占有率の改善度

(a) 多角形 18 個

	占有率			
	ビンパッキング	BSG コンパクション	追加コンパクション	全改善度
ランダム順	0.702648	0.726874 (0.024226 改善)	0.755659 (0.028785 改善)	0.053011
降順ソート順	0.822328	0.822328	0.822328	0
昇順ソート順	0.749118	0.749118	0.769003 (0.019885 改善)	0.019885

(b) 多角形 33 個

	占有率			
	ビンパッキング	BSG コンパクション	追加コンパクション	全改善度
ランダム順	0.728667	0.798690 (0.070023 改善)	0.829552 (0.030862 改善)	0.100885
降順ソート順	0.845660	0.845660	0.845660	0
昇順ソート順	0.845660	0.845660	0.845660	0

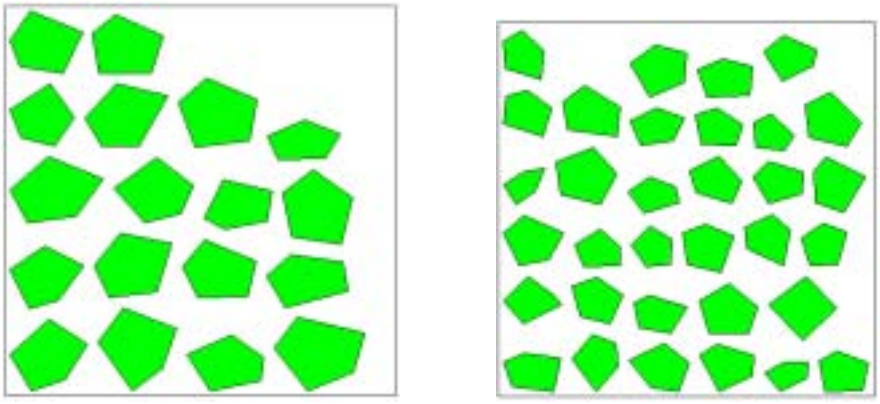
第4章　多角形の配置調整

前章までは, 一定枠に対する長方形の占有率を高くする事だけに着目してきた. 本章では, 長方形に近似していた凸多角形を復元し宣伝用広告チラシに応用する事を考える. 宣伝用広告チラシは, 一部分に多角形が偏りすぎていると見栄えが良くない. そこで, 多角形の空き領域が一定になるよう配置調整を行なう. 配置調整を行なうには, 各多角形の回りにどのくらいの領域があるのか知る必要がある. その方法として, 多角形間におけるボロノイ図とその双対関係にあるデローネイ三角形分割の概念を利用する. 具体的には, 各多角形の辺で定義される線分集合に対するデローネイ三角形分割を求め, 空き領域を見出す. 調整手法としては, 包含円を用いた手法とミンコフスキー和を用いた手法の 2 つを用いる.

4.1　凸多角形への復元

長方形に近似していた凸多角形を復元する. 長方形を凸多角形に復元した図を, 図 4.1(ランダム順), 図 4.2(降順ソート順), 図 4.3(昇順ソート順) に示す.

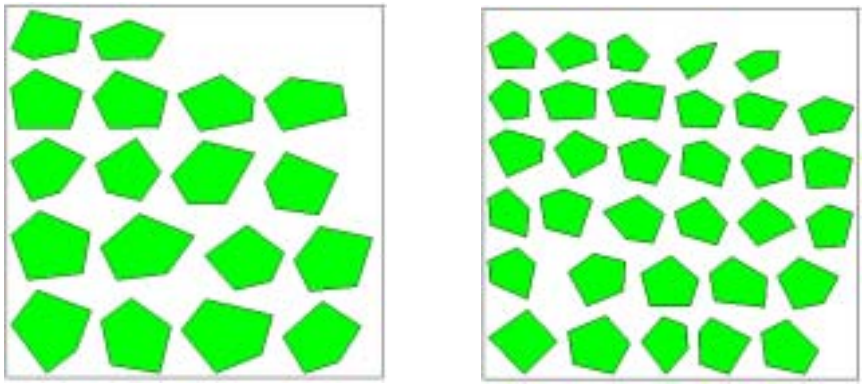
ランダム順入力



(a) 多角形 18 個　　(b) 多角形 33 個

図 4.1: 凸多角形への復元 (ランダム順)

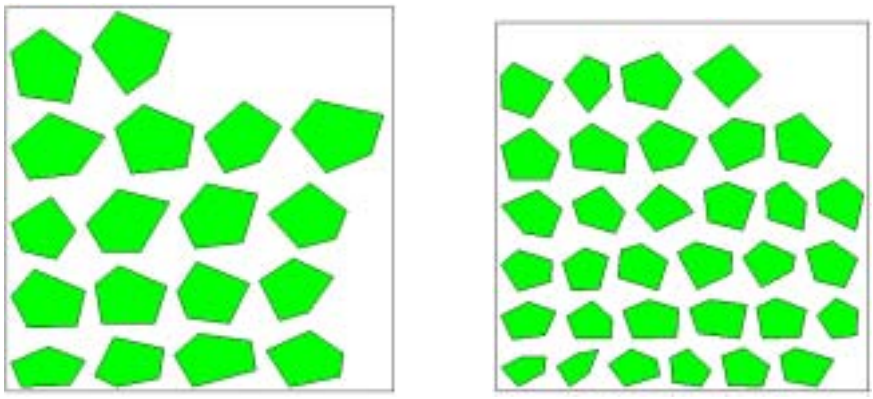
降順ソート順入力



(a) 多角形 18 個　　(b) 多角形 33 個

図 4.2: 凸多角形への復元 (降順ソート順)

昇順ソート順入力



(a) 多角形 18 個　　(b) 多角形 33 個

図 4.3: 凸多角形への復元 (昇順ソート順)

4.2　見栄えの良い多角形配置とは

見栄えの良い多角形配置は, 人間の主観に依存する所が大きい. なぜなら, 人により美しさを判断する基準が異なる為である. 従って客観的に美しい多角形配置を定義する事は困難である. 本研究では, 多角形間の空き領域が均一であるような配置を見栄えが良い事とした.

4.3　包含円を用いた配置調整

4.3.1　空き領域の発見

配置調整を行なうには多角形の回りにどれくらいの空きスペースがあるか知る必要がある. そこで各多角形間のデローネイ三角形分割を計算し, 多角形の周囲に接続する三角形を列挙し空きスペースを見出す. 図 4.4 に多角形を取り囲む領域を示す.

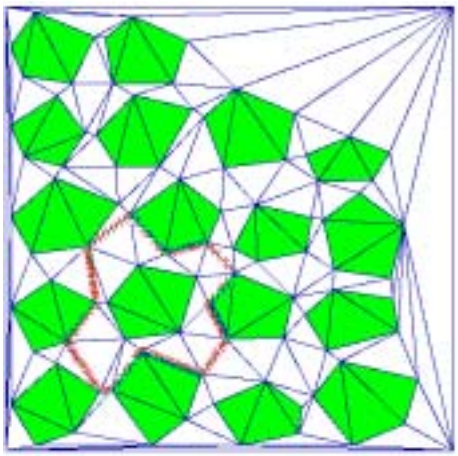


図 4.4: 多角形を取り囲む領域

4.3.2　多角形の移動

多角形を移動させるには, どの多角形を移動させるのかを決める必要がある. そこで, 各多角形の包含円とそれを取り囲む領域の最大内接円を求める. 図 4.5 に, 多角形の包含円とそれを取り囲む領域の最大内接円を示す.

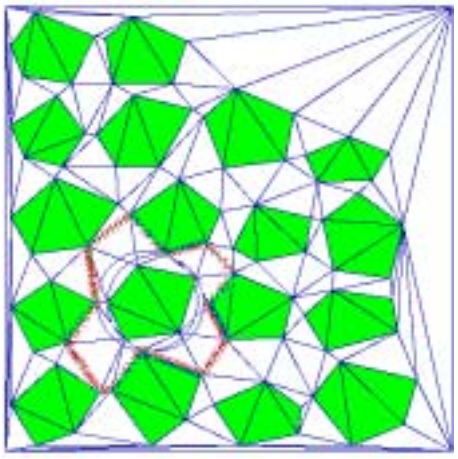
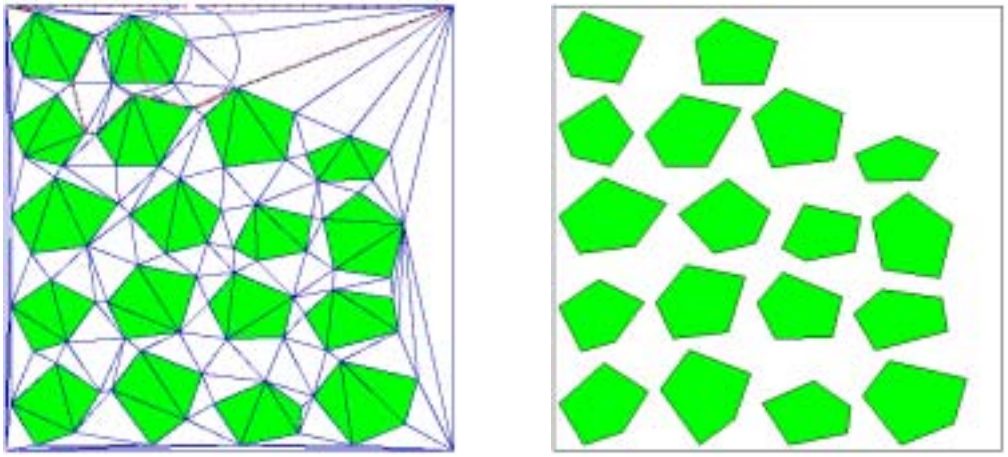


図 4.5: 多角形の包含円と, 多角形を取り囲む領域の最大内接円

全ての多角形の包含円と, 多角形を取り囲む領域の最大内接円の距離を計算し, 最終的に 2 つの円の距離が最も大きい多角形を, 最大内接円の中心に移動させる. その様子を図 4.6 に示す.



(a) 移動多角形の決定　　(b) 移動後の多角形

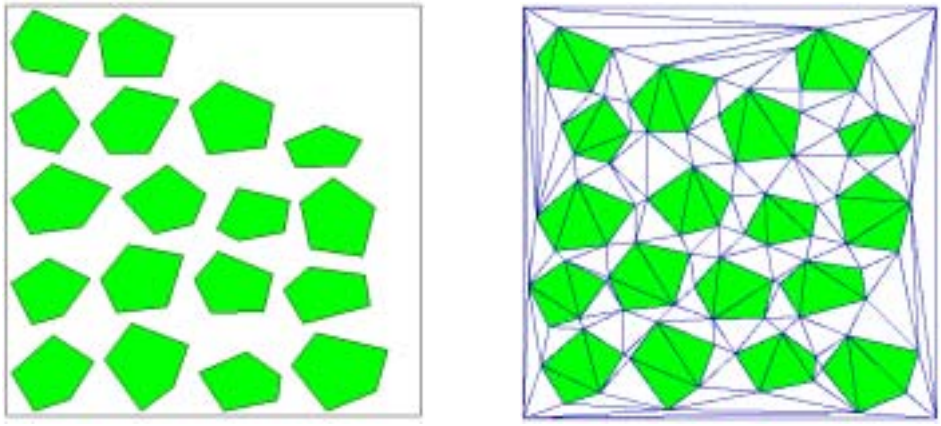
図 4.6: 多角形の移動

これらの作業を, ある程度収束するまで繰り返し終了する.

4.3.3 配置調整結果

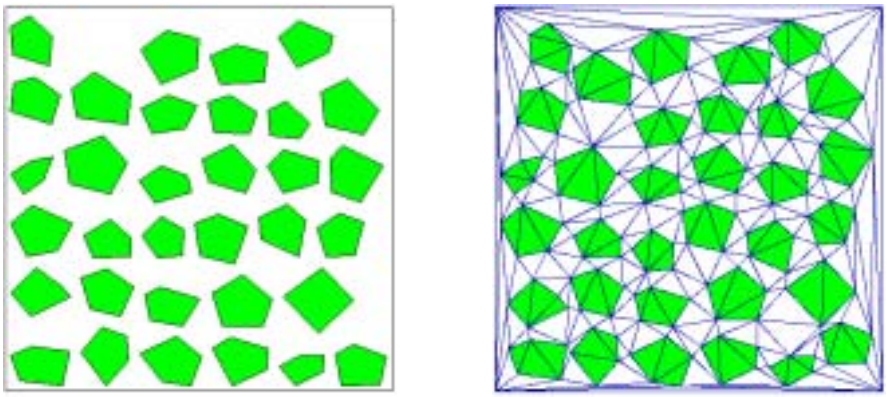
多角形の配置調整前と調整後の結果を, ランダム順入力は図 4.7, 図 4.8 に, 降順ソート順入力は図 4.9, 図 4.10 に, 昇順ソート順入力は図 4.11, 図 4.12 に示す.

ランダム順入力



(a) 調整前　　(b) 調整後

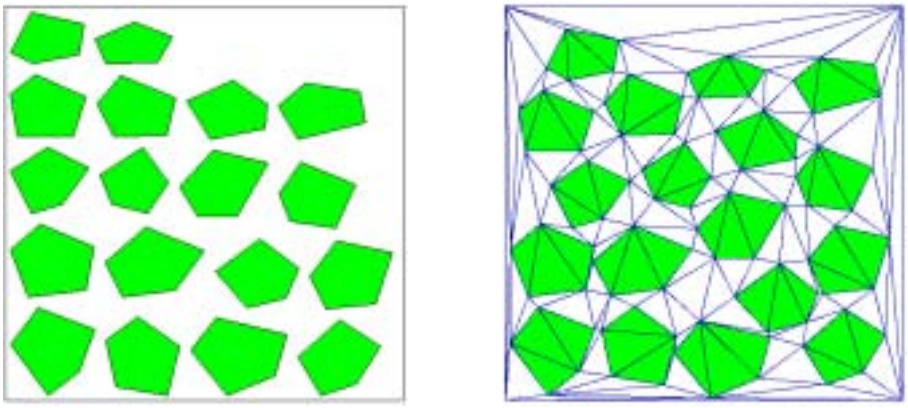
図 4.7: 包含円を用いた配置調整結果 (多角形 18 個　ランダム順)



(a) 調整前　　(b) 調整後

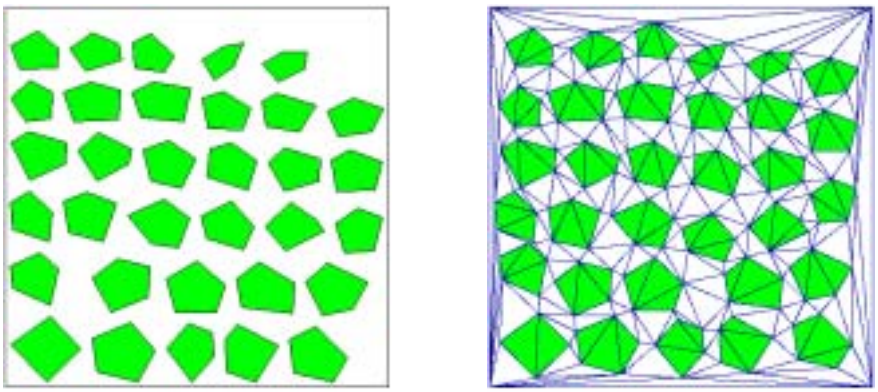
図 4.8: 包含円を用いた配置調整結果 (多角形 33 個　ランダム順)

降順ソート順入力



(a) 調整前　　(b) 調整後

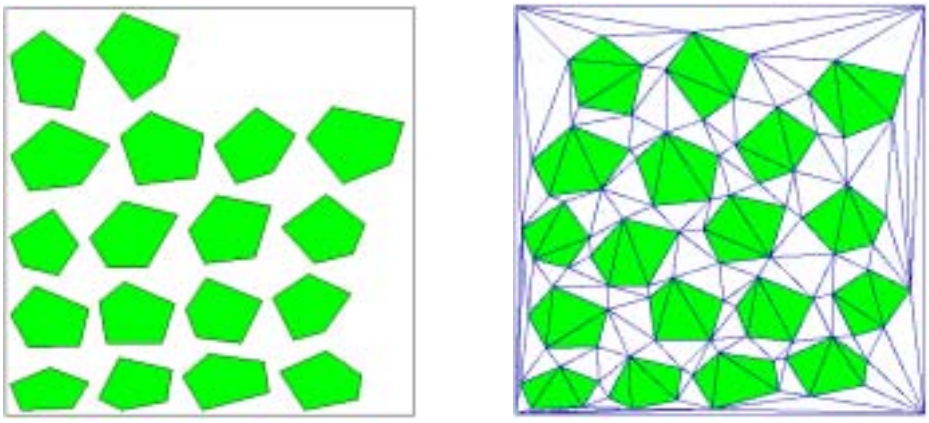
図 4.9: 包含円を用いた配置調整結果 (多角形 18 個　降順ソート順)



(a) 調整前　　(b) 調整後

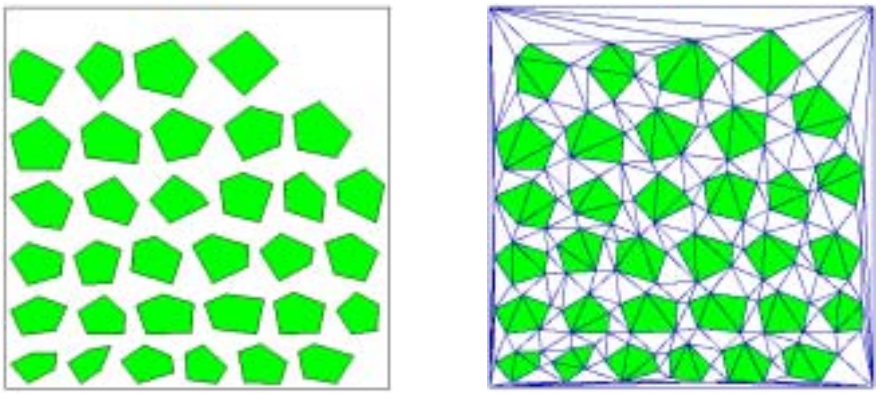
図 4.10: 包含円を用いた配置調整結果 (多角形 33 個　降順ソート順)

昇順ソート順入力



(a) 調整前　　(b) 調整後

図 4.11: 包含円を用いた配置調整結果 (多角形 18 個　昇順ソート順)



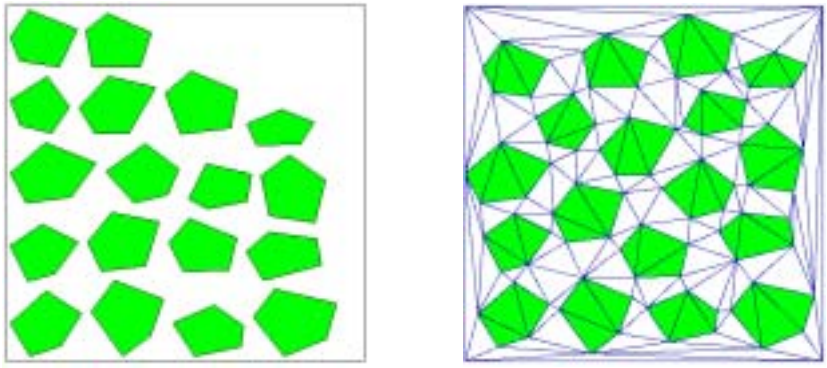
(a) 調整前　　(b) 調整後

図 4.12: 包含円を用いた配置調整結果 (多角形 33 個　昇順ソート順)

4.3.4　配置調整の高速化

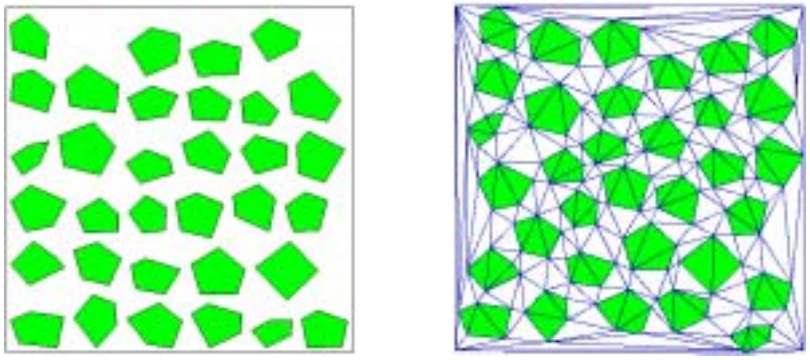
今まで包含円を用いた配置調整は, 全ての多角形の移動距離を求め最大距離をもつ多角形を移動させていた為, 移動多角形を見出すのに時間がかかっていた. そこで全ての多角形の移動距離を求めるのではなく, 配置された順番に多角形の移動距離を求め, 移動可能なら逐次移動させていく事とする. これにより計算時間短縮と, 新たな多角形配置を目指す. 多角形を移動させ配置調整した結果を, ランダム順入力は図 4.13, 図 4.14 に, 降順ソート順入力は図 4.15, 図 4.16 に, 昇順ソート順入力は図 4.17, 図 4.18 に示す.

ランダム順入力



(a) 調整前　　(b) 調整後

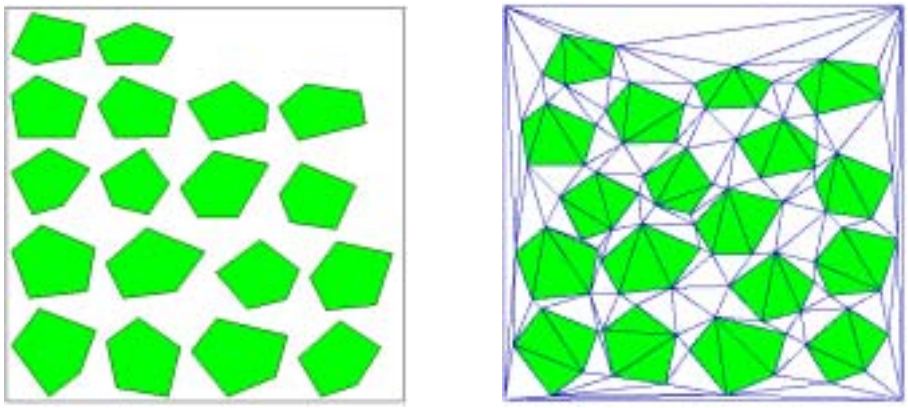
図 4.13: 逐次移動させた配置調整結果 (多角形 18 個　ランダム順)



(a) 調整前　　(b) 調整後

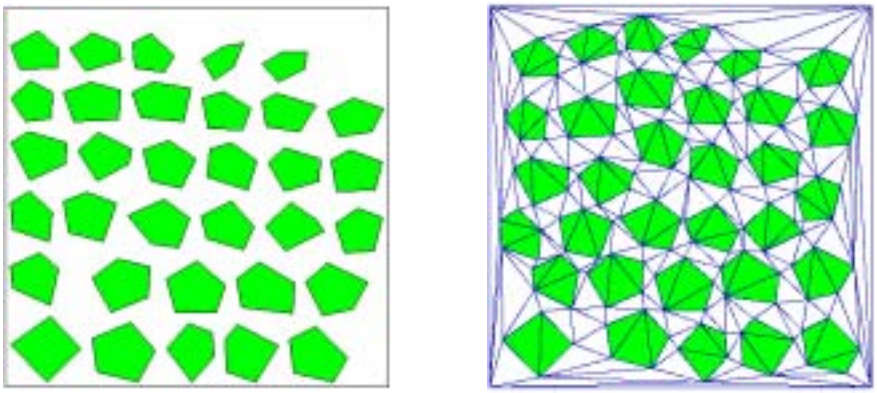
図 4.14: 逐次移動させた配置調整結果 (多角形 33 個　ランダム順)

降順ソート順入力



(a) 調整前　　(b) 調整後

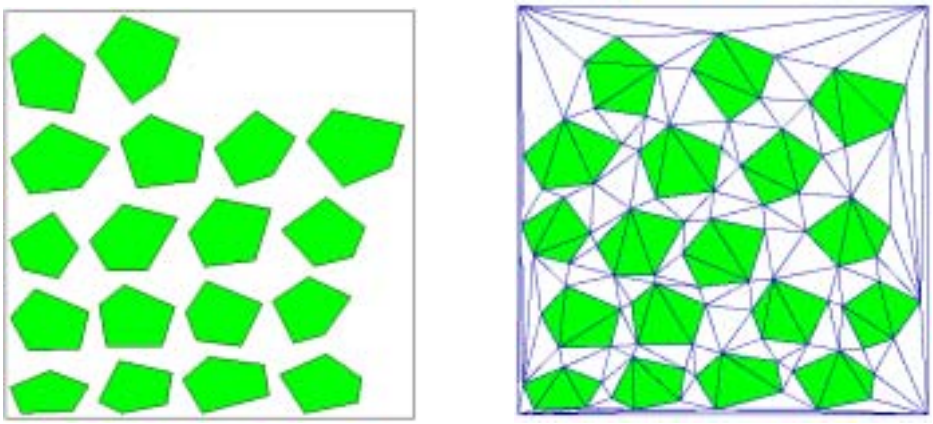
図 4.15: 逐次移動させた配置調整結果 (多角形 18 個　降順ソート順)



(a) 調整前　　(b) 調整後

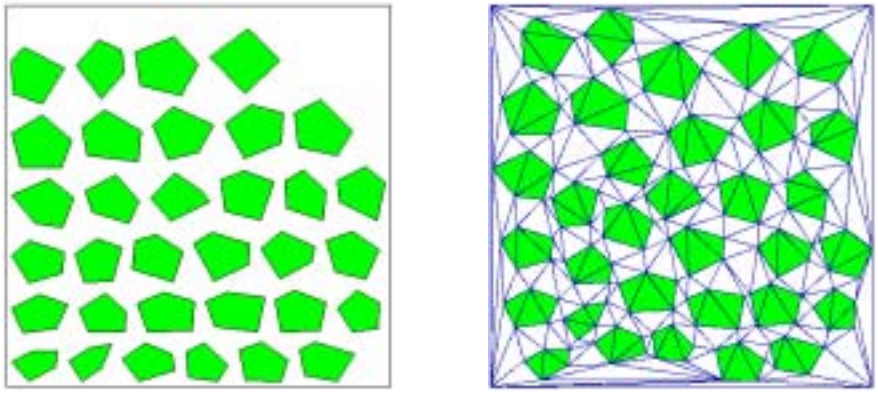
図 4.16: 逐次移動させた配置調整結果 (多角形 33 個　降順ソート順)

昇順ソート順入力



(a) 配置前　　(b) 配置後

図 4.17: 逐次移動させた配置調整結果 (多角形 18 個　昇順ソート順)

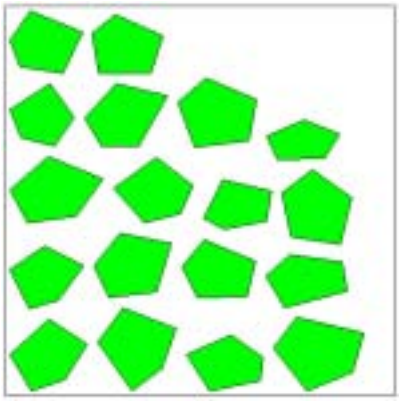


(a) 配置前　　(b) 配置後

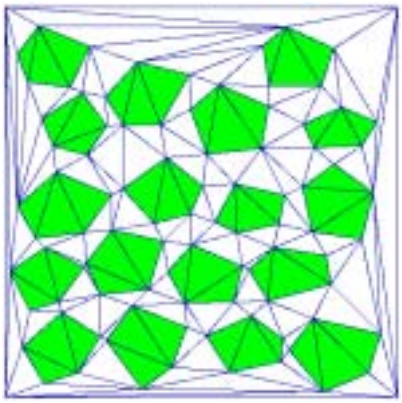
図 4.18: 逐次移動させた配置調整結果 (多角形 33 個　昇順ソート順)

4.3.5　評価

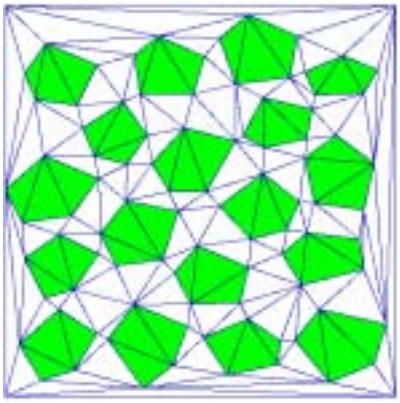
多角形配置に関しては, 全ての結果において空き領域が均一になり見栄えが良くなったといえる. 例えば図 4.19 を見てみる. 図 4.19 は, 多角形 18 個をランダム順に並べ調整した図 4.7 と図 4.13 をまとめたものである. 図 4.19 を見ると, 調整前は右上に大きな空き領域があったが, 調整後は大きな空き領域がなくなり改善された. また, 図 4.19(c) は, 計算時間の短縮を目指し多角形を配置順に逐次移動させて調整した結果であるが, 図 4.19(b) と同じく空き領域が均一となり実用的である事が分かった.



(a) 調整前



(b) 最大移動距離の多角形を移動させて配置調整



(c) 多角形を配置順に逐次移動させて配置調整

図 4.19: 配置調整の評価 (多角形 18 個　ランダム順)

4.4　ミンコフスキー和を用いた配置調整

前節では包含円を用いた配置調整を実施した. 本節ではミンコフスキー和の概念を用いて配置調整を実施する.

4.4.1　包含円を用いた配置調整手法との違い

最初に以下の図 4.20 を見てもらいたい.
図 4.20 を見ると, 右上に大きな空き領域があり, 左下に比較的小さい多角形があるのが分かる. 左下の多角形を, 大きな空き領域に移動させる事は可能だろうか. 前節での包含円を用いた配置調整手法では, 多角形の移動は不可能である. なぜなら前節の配置調整手法では, 多角形をずらして配置調整を実施している為, 周囲の多角形を飛び越えて移動させる事ができないからである. しかしミンコフスキー和の概念を用いると, 周囲の多角形を飛び越えて直接大きな空き領域へ移動させる事が可能である. 従って, より効果的な配置調整が可能となる.

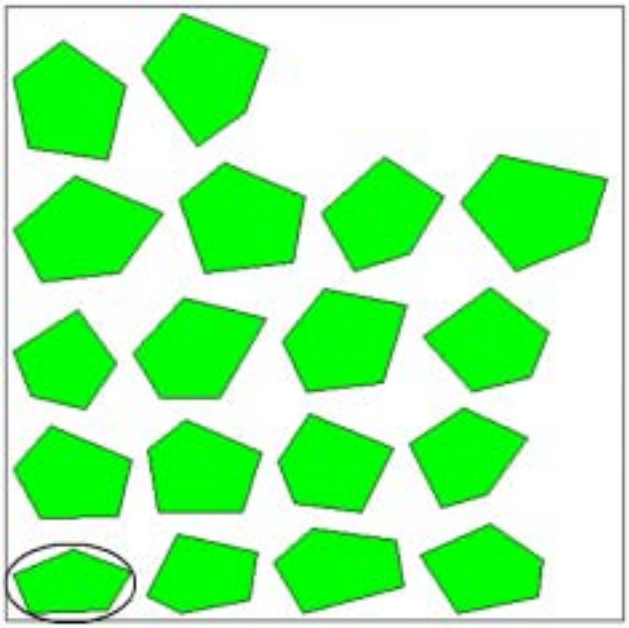
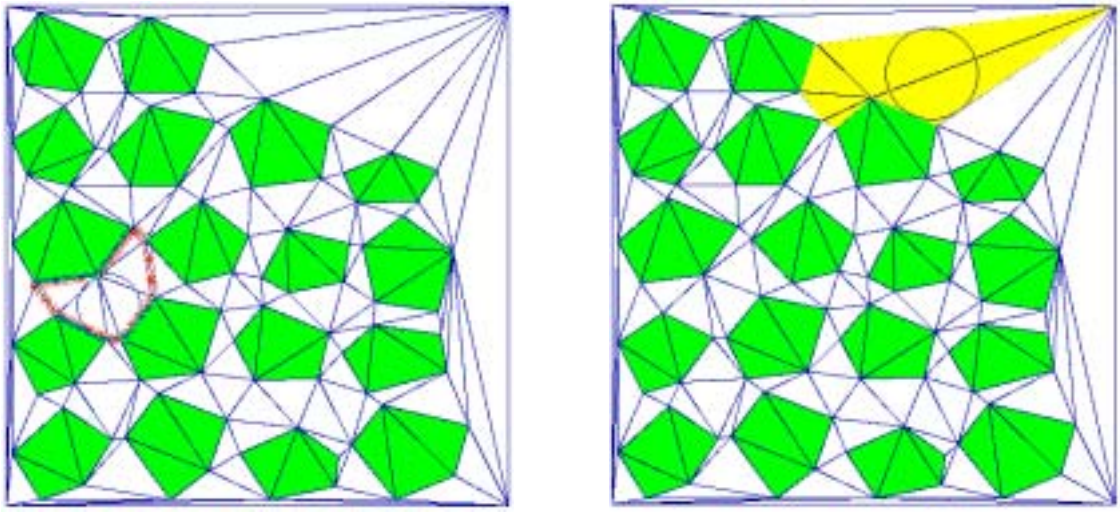


図 4.20: 左下の多角形は移動可能か?

4.4.2　空き領域の発見と移動領域の決定

多角形を移動させるには, 空き領域を見出す必要がある. 空き領域は前節までと同じく各多角形間のデローネイ三角形分割を計算して見出す. 包含円を用いる手法では多角形の周囲の空き領域を見出していたが, 本手法では多角形の頂点を取り囲む空き領域を見出す. 図 4.21(a) に, 空き領域の１つを示す. 全ての空き領域を見出し, その領域の内接円を計算する. そして全ての内接円の中で最も直径が大きい領域を, 移動させる領域とする. 図 4.21(b) に内接円が最大となる移動領域を示す.



(a) 移動領域の候補　　(b) 決定した移動領域

図 4.21: 多角形の移動領域

4.4.3 多角形移動アルゴリズム

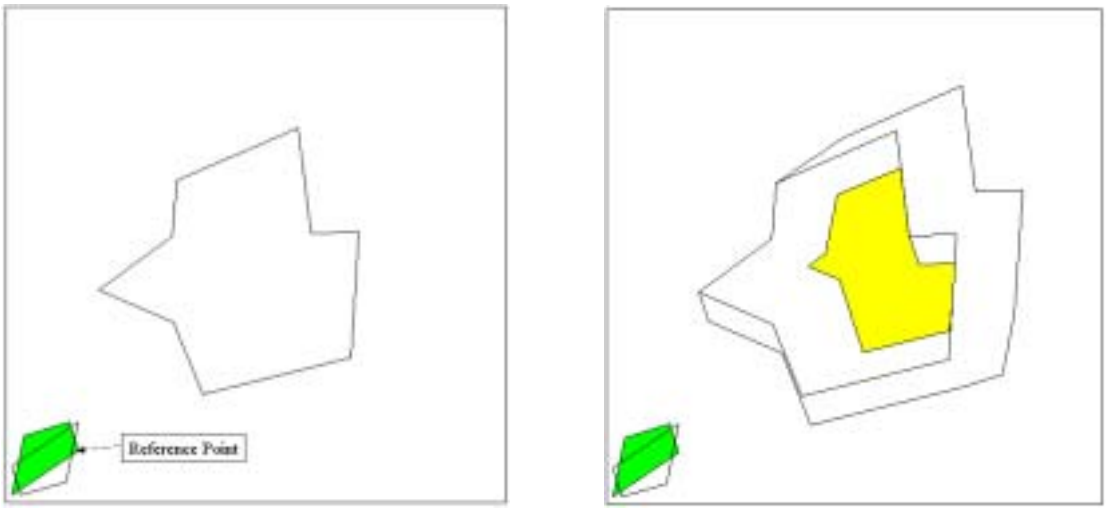
ミンコフスキー和を用いた配置調整は, 移動する領域が非凸多角形であるので, 非凸多角形の中に凸多角形を詰め込む問題として扱う事ができる. 凸多角形を N, 非凸多角形を NC とすると, ミンコフスキー和 $NC \oplus N$ は, 式 (4.1) で表す事ができる.

$$NC \oplus N = \{(x, y) | N(x, y) \cap NC \neq 0\} \tag{4.1}$$

ミンコフスキー和を用いた多角形移動は, 以下の $step1$ から $step3$ の手順で行なう.

step1 ミンコフスキー和の計算

凸多角形を非凸多角形の中に詰め込む場合, 上下左右逆転したミンコフスキー和を計算する事になる. そこで, 180 度回転させた凸多角形と非凸多角形とのミンコフスキー和を計算する. 図 4.22(a) に, 凸多角形と非凸多角形を示す. 図 4.22(a) で, 塗りつぶされていない凸多角形が 180 度回転された凸多角形である. ミンコフスキー和を計算し, もし非凸多角形の内側にミンコフスキー和が存在しない領域がある時, その領域が移動領域となる. 図 4.22(b) に移動領域を示す. 非凸多角形内の塗りつぶされた部分が移動領域である.



(a) 凸多角形と非凸多角形　　(b) ミンコフスキー和

図 4.22: 移動領域の計算

step2 移動領域に内接円を描く

非凸多角形内部の移動領域に内接円を描く. 図 4.23 に移動領域内の内接円を示す.

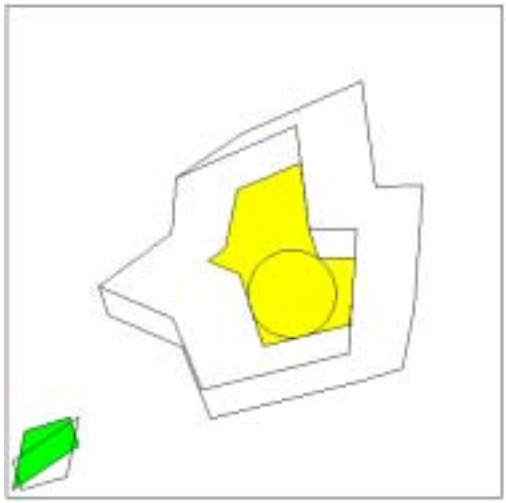


図 4.23: 移動領域の内接円

step3 多角形の移動

内接円の中心に凸多角形の参照点を移動させる. 図 4.24 に, 移動領域に凸多角形を移動させた図を示す.

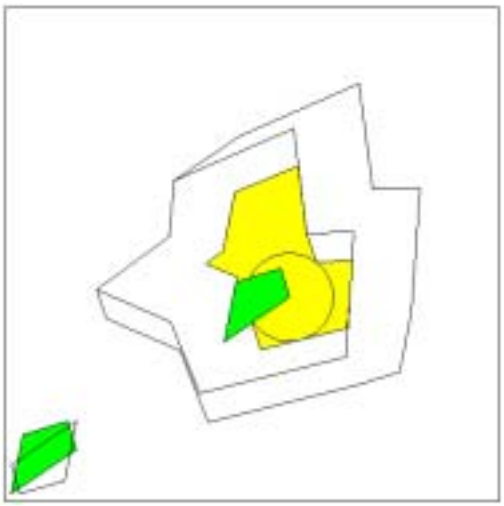


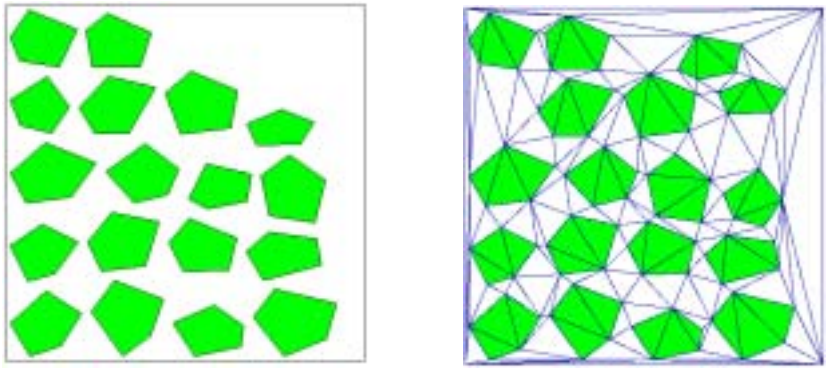
図 4.24: 多角形の移動

4.4.4　実験結果

本研究では, 全ての凸多角形と空き領域である非凸多角形のミンコフスキー和を計算し, その中で移動領域の内接円が最大である凸多角形を移動させる. また, 連続して同じ多角形が移動するのを避けるため, 一度移動した多角形は, 次には移動させないようにする. そしてこの作業を何度か繰り返し終了する.

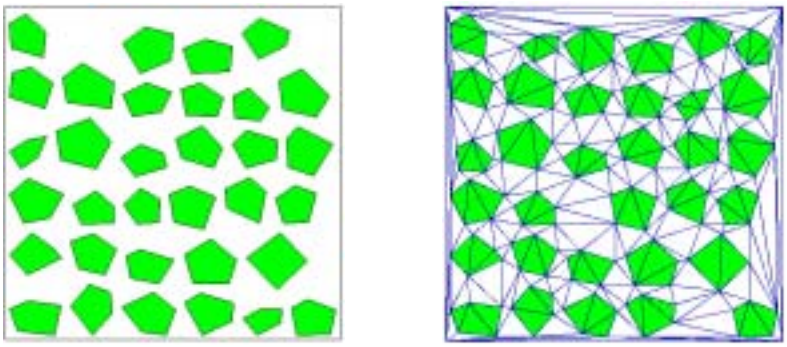
この手法を用いて配置調整を行なった結果を, ランダム順入力は図 4.25, 図 4.26 に, 降順ソート順入力は図 4.27, 図 4.28 に, 昇順ソート順入力は図 4.29, 図 4.30 に示す.

ランダム順入力



(a) 調整前　　(b) 調整後

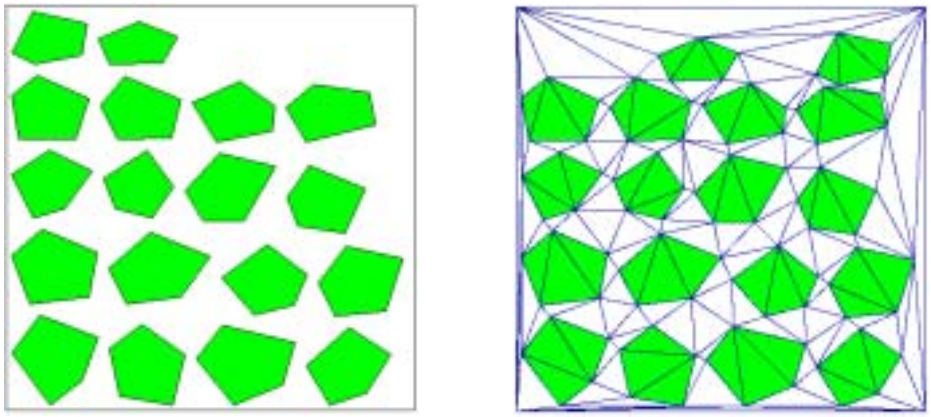
図 4.25: ミンコフスキー和を用いた配置調整結果 (多角形 18 個　ランダム順)



(a) 調整前　　(b) 調整後

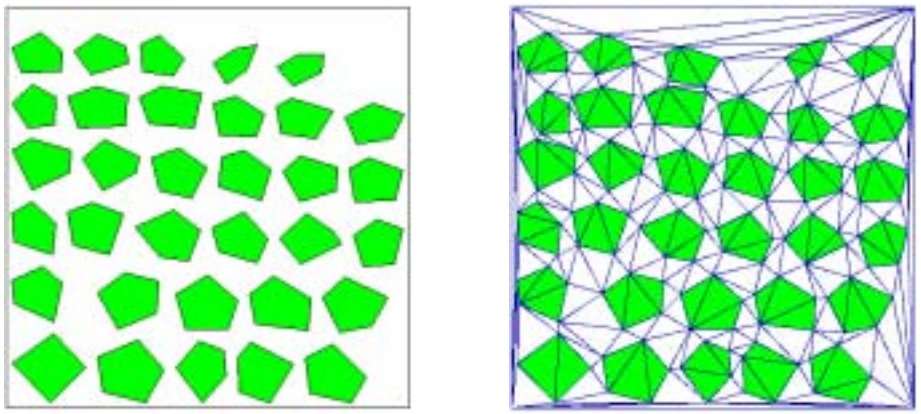
図 4.26: ミンコフスキー和を用いた配置調整結果 (多角形 33 個　ランダム順)

降順ソート順入力



(a) 調整前　　(b) 調整後

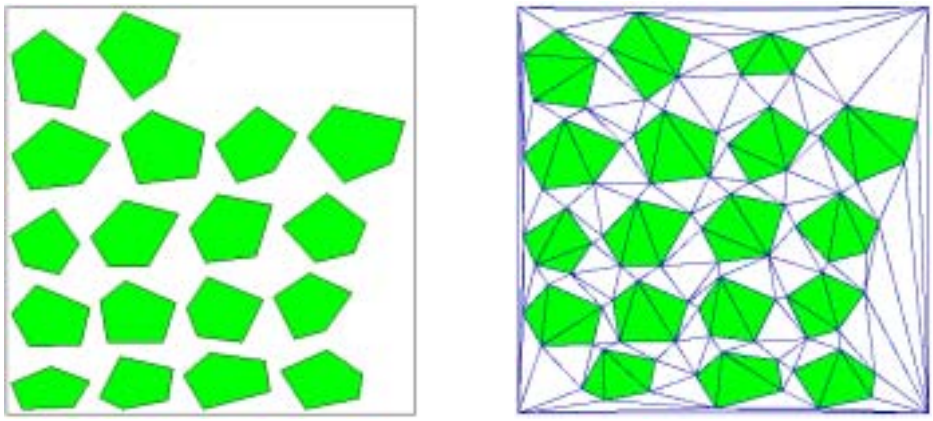
図 4.27: ミンコフスキー和を用いた配置調整結果 (多角形 18 個　降順ソート順)



(a) 調整前　　(b) 調整後

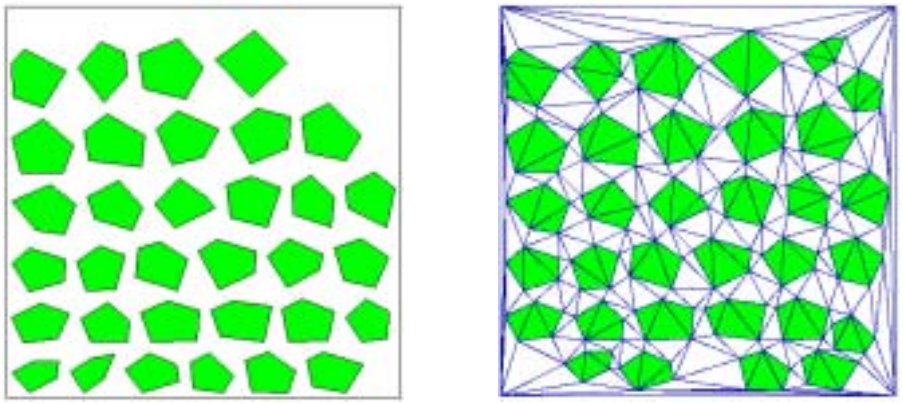
図 4.28: ミンコフスキー和を用いた配置調整結果 (多角形 33 個　降順ソート順)

昇順ソート順入力



(a) 調整前　　(b) 調整後

図 4.29: ミンコフスキー和を用いた配置調整結果 (多角形 18 個　昇順ソート順)



(a) 調整前　　(b) 調整後

図 4.30: ミンコフスキー和を用いた配置調整結果 (多角形 33 個　昇順ソート順)

4.4.5 評価

ミンコフスキー和を用いた配置調整は, 基本的に最も小さい多角形を最大空き領域に移動させる手法である. また最大空き領域に関しては, 枠内の上方に存在する場合が多い. 昇順ソート順に配置された図 4.29, 図 4.30 を見てみると, 大きい多角形は上方に, 小さい多角形は下方に配置されている事から, 多角形が大きく移動し, バランスの良い配置結果を得る事ができた. 反対に, 降順ソート順に配置された図 4.27, 図 4.28 を見てみると, 大きい多角形は下方に, 小さい多角形は上方に配置されている事から, 多角形の移動が小さく, あまり良い結果にはならなかった. 従って降順ソート順に関しては, 包含円を用いて配置調整を実施する方が適していると考えられる. ランダム順に関しては, 図 4.25 はそこそこバランスの良い配置が得られたものの, 図 4.26 は元々空き領域が小さい事もありあまり変化がなかった.

計算時間に関しては, 包含円を用いる手法より多くかかった. なぜなら空き領域を見出す事に関して, 包含円を用いる手法は, 多角形周辺の領域を計算しただけであったが, ミンコフスキー和を用いる手法は, さらに多角形の頂点周辺の全ての領域を計算したので, 頂点の数だけ余計に時間がかかった為である.

第5章　まとめ

5.1　実験結果に基づく評価

長方形の初期配置に関しては, 長方形を降順または昇順ソートして配置していくと, 占有率の点でランダム順に配置する時と比べて優れる傾向がある事が分かった. だがランダム順に配置したとしても, 配置後に 2 次元コンパクションを実施する事により, 占有率をある程度まで高められる可能性がある事が分かった.

多角形の配置調整に関しては, 包含円を用いた手法とミンコフスキー和を用いた手法の 2 つを行なったが, 両者共に空き領域がほぼ均一となり実用に耐えられる結果が得られた. 包含円を用いた手法は, 空き領域を見出すのに時間がかかるものの, 空き領域が均一となる見栄えの良い配置が実現できた. また計算時間を短縮する為, 多角形を配置順に逐次移動させる手法を実施したが, 十分実用にたえる配置を実現する事ができた. ミンコフスキー和による手法は, 包含円を用いた手法では不可能だった周囲の多角形を飛び越えての移動が可能となり, よりバランスのよい配置が可能となった. 例えば, 昇順ソート順で配置した時のように小さな多角形が下方に密集した場合においても, 上方の大きな空き領域に多角形を移動させる事により, バランスの良い配置を実現する事ができた.

最後に本研究で実施した配置調整手法は, 多角形を調整する前に比べて見た目の印象は確実に良くなった. 従って調整手法としては十分な成果を得る事ができた. だが 2 つの配置手法にはそれぞれ長所もあれば短所もある. 短所についての改善は今後の検討課題となり, 次節で述べる.

5.2　今後の課題

本研究で実施した, 包含円を用いる手法も, ミンコフスキー和を用いる手法も, 多角形を移動させる上でそれぞれ長所や短所があった. 短所をあげると, 包含円を用いる手法は周囲の多角形を飛び越えて大きく移動できない事であり, ミンコフスキー和を用いる手法は, 降順ソート順に配置された多角形配置には, あまり効果がなかった事である. この 2 つの手法を状況に応じて使い分けるシステムを構築し互いの短所を補う事ができれば, さらに計算時間短縮と見栄えの良い配置が得られるものと思われる.

また本研究では, 多角形の配置調整を行なう際, 1 度に 1 個の多角形を考慮して移動を行なった. だが同時に複数の多角形を考慮し移動させる事ができれば, さらに見栄えの良い多角形配置が得られるものと思われる.

謝辞

本研究を進めるにあたり, 研究のきっかけを与え, アルゴリズムやプログラミングなど多くの事を熱心に指導してくださった浅野哲夫教授に深く感謝いたします. また, アルゴリズムゼミや文献ゼミ, 研究室生活において多大なお世話になった中野浩嗣助教授, 小保方幸次助手に深く感謝いたします. 最後に公私にわたりお世話になった研究室の先輩, 同級生, 後輩ならびに友人の皆様に深く感謝いたします. ありがとうございました.

関連図書

[1] Naveed Sherwani: "Algotrithms for VLSI Physical Design Automation THIRD EDITION", Kluwer Academic Publishers, pp.191-218, 1999.

[2] Thomas Lengauer: "Combinatorial Algorithms for Integrated Circuit Layout", B.G.Teubner, pp.328-377, 1990.

[3] Victor J.Milenkovic: "Rotational Polygon Overlap Minimization", Proc. Thirteenth Annual ACM Symposium on Computational Geometry, pp.334-343, 1997.

[4] Victor J.Milenkovic: "Densest Translational Lattice Packing of Non-Convex Polygons", Proc. Computational Geometry 2000 Hong Kong China, pp.280-289, 2000.

[5] Victor J.Milenkovic: "Compaction and Separation Algorithms for Non-Convex Polygons and Their Applications", Proc. European Journal of Operationd Research, pp.539-561, 1995.

[6] S.Nakatake, H.Murata, K.Fujiyoshi, and Y.Kajitani, " Module placement on BSG-structure and IC layout applications ", International Conf.on Computer Aided Design, pp.484-491,San Jose, US, Nov 1996.

[7] S.Nakatake, H.Murata, K.Fujiyoshi, and Y.Kajitani, " Module packing based on the BSG structure and IC layout applications ", IEEE Trans. CAD, vol.17, no.6, p.519-529, June 1998.