



計算幾何学  
Computational Geometry


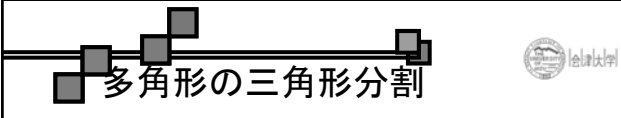


第七章 多角形の三角形分割  
Polygon Triangulation



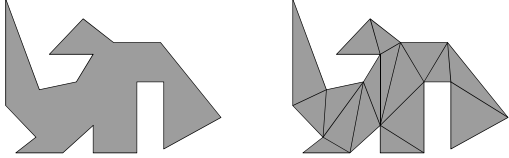

内容

- 定義
  - 多角形の三角形分割
- 問題の提起
  - 美術館監視→美術館問題
- 三角形分割のプロセス
  - 単純多角形 (simple polygon)
  - 単調多角形 (monotone polygon)
  - 三角形 (triangle)

多角形の三角形分割

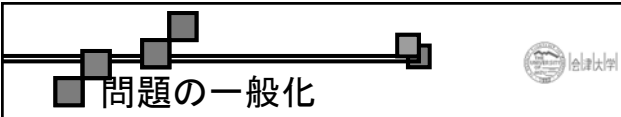
- $P$ を $n$ 頂点の単純な多角形とする
- $P$ の三角形分割
  - 交差しない対角線の極大集合によって、多角形を三角形に分割したもの
  - 対角線:  $P$ の2頂点を $P$ の内部だけを通して結ぶ開線分


美術館問題(art gallery problem)

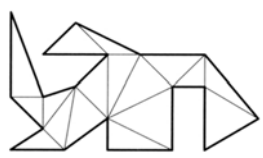

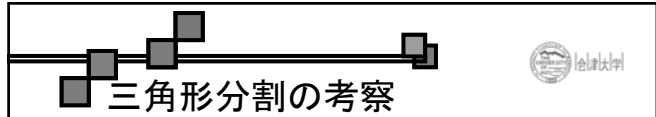
- 全館貴重な展示品を漏れなく監視するには  
→必要なカメラ台数?





問題の一般化

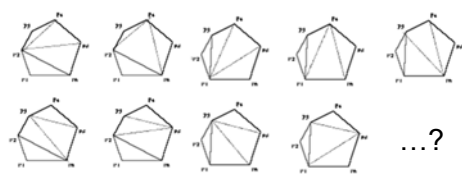

- 多角形 $P$ の内に、最小個数の点集合 $G$ を選び、 $P$ 内のどの点も $G$ の少なくとも一つの点から見えるようにする
- $P$ 内の2点が互いに見える(visible)→この2点を結ぶ線分の全体が $P$ 内にある
- $P$ を三角形に分割
  - 一つ三角形領域に一台カメラ→全域漏れなく見える

三角形分割の考察

- 多角形を対角線で区切り、三角形に分割。
- 対角線同士は互いに交わらないように。

1. 三角形分割は常に存在する? →Y
2. 多角形に何個の三角形がある? → $n-2$
3. 一意に決まらない→何通りある? → $_{2n}C_n/(n+1)$

### 単純多角形(simple polygon)

- 互いに**交差しない**閉じた多辺連続チェーンに囲まれた領域
  - 平面を2つの互いに素な領域に分割する
  - 有界な内部領域 + 非有界な外部領域
  - 有界領域内に穴がない

Simple polygons

Non-simple polygons

### 対角線と三角形分割

- 対角線(diagonal)
  - 単純多角形の内部を通って、2頂点を結ぶ開線分
- 三角形分割(triangulation)
  - 交差しない対角線の極大集合(?)によって多角形を三角形に分割すること

### 定理1

- どんな単純多角形も三角形分割が可能
- $n$ 個の頂点を有する単純多角形の任意の三角形分割にはちょうど $n-2$ 個の三角形がある

考察: カメラ設置位置と台数

- 三角形内  $\rightarrow n-2$
- 対角線上  $\rightarrow n/2$
- 頂点上  $\rightarrow n/3$

### 3-彩色(3-coloring)

- 単純多角形の各頂点に白、灰、黒のいずれかの色を割り当てる。但し、隣接の2頂点は必ず異なる色を付ける
- どの三角形も白・灰・黒の頂点を一つずつ持つ
- 何れの色のみカメラを配置すれば、多角形全体の監視が可能
- 高々 $\lfloor n/3 \rfloor$ 台のカメラが十分

### 最悪の例

- 頂点に配置されたカメラの監視範囲は接続している三角形に限らない  $\rightarrow$  改善の余地?
- $n/3$ 台以下のカメラで任意の多角形を監視する可能性は?
- 答え  $\Rightarrow$  No

$\lfloor n/3 \rfloor$  prongs 櫛状の多角形

### 定理2 (美術館定理)

- $n$ 個の頂点を有する単純多角形に対して、多角形内の任意の点が少なくとも一台のカメラから見えるようにするのに、 $\lfloor n/3 \rfloor$ 台のカメラが必要になることがあり、またその台数が常に十分である
- $n/3$ 台は必要なカメラ台数の上限
- 言い換えれば、 $n/3$ 台以上を必要とする可能性はない。 $n/3$ 台以下で監視できる時もある
- 充分条件であるが、必要条件ではない

### ■ 三角形分割のプロセス

単純多角形 (simple polygon) → 単調多角形 (monotone polygon) → 三角形 (triangle)

### ■ y単調多角形の特徴

- y軸と垂直な直線 (水平線) に対して、多角形と交差する部分の線分が**連結**である
- 最も上の頂点から最も下の頂点まで、左側又は右側の境界の多辺連続チェーンをたどる時、その走行方向は常に**下向き**又は**水平方向**だけ⇒**変曲点なし**
- 変曲点 = 向き変化発生点

### ■ 5種類の頂点

頂点種類	左右隣接2頂点	内角
□ 出発点	全て下方	$< \pi$
■ 最終点	全て上方	$< \pi$
● 普通点	上・下方各一点	$< \pi$ $> \pi$
▲ 分離点	全て下方	$> \pi$
▼ 統合点	全て上方	$> \pi$

### ■ y単調な多角形を求める

- 多角形の最も上の頂点から最も下の頂点へたどっていくとき、上下向きが変化する頂点 (**変曲点**) を取り除けば良い
- 取り除く⇒変曲点から**対角線**を引く

### ■ 単純多角形 → 単調多角形

- 変曲点 (turn vertex)
  - 多角形の境界をたどる時、向きが変化する頂点
    - 出発点、最終点
    - 分離点、統合点
- y単調に分割の目的
  - 変曲点の除去 (分離点と統合点)
- y単調に分割の方法
  - 対角線を添加

### ■ y単調多角形の性質

- 分離点も統合点も持たない多角形。
- 論題: y単調でないPは分離点か統合点を持つ。
- 証明:
  1. Pはy単調でないので、Pと交差する部分が2つ以上の線分に分かれるような水平線lが存在する。
  2. このとき最も左の線分の左端点をp, 右端点をqとする。
  3. qから出発して、Pの内部を左に見ながらPの境界線を辿ると、ある点rで境界線はlと再び交差するはずである。

証明(続き)

4.  $p \neq r$  のとき

- $q$  から  $r$  までの道のりで最も高いところにある頂点は分離点でなければならない。

5.  $p = r$  のとき

- $q$  から出発して、 $P$  の内部を右に見ながら  $P$  の境界線を辿る。このとき境界線が  $l$  と交差する点を  $r'$  とする。
- もし  $r' = p$  なら、 $P$  の境界線は  $l$  と2回しか交差しないことになるため、 $r' \neq p$  である。
- このとき  $q$  から  $r'$  までの道のりで最も低いところにある頂点は、統合点でなければならない。

### 分離点と統合点の除去

- 分離点から上の頂点に向かう対角線を引く
- 統合点から下の頂点に向かう対角線を引く
- 対角線の具体的な引き方？

### 分離点の処理

■ 仮想的な走査線  $l$  を上から下へと動かしていく。

■  $l$  が分離点  $v_i$  に到達したとき、 $l$  からの距離が最小で  $v_i$  より上にある頂点と、 $v_i$  とを結ぶ。

↓

辺  $e_i$  のヘルパー

### 統合点の処理

■  $l$  が統合点  $v_i$  に到達したとき、 $l$  からの距離が最小で  $v_i$  より下にある頂点と、 $v_i$  とを結ぶ。

↓

$v_i$  の次に  $e_j$  のヘルパーになる頂点

↓

位置  $= l_1 \Rightarrow \text{helper}(e_j)$

↓

位置  $= l_2 \Rightarrow \text{helper}(e_j)$

### 平面走査法

基本的な考え方

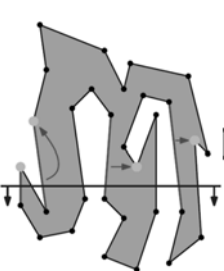
- $v_1, v_2, \dots, v_n$  逆時計回り順に並べた頂点
- $e_1, e_2, \dots, e_n$  辺集合、 $e_i = \overrightarrow{v_i v_{i+1}}$ 、 $e_n = \overrightarrow{v_n v_1}$
- イベント点: 多角形の頂点
- 目的:
  1. 各分離点からその上にある頂点へ対角線を引く
  2. 各統合点からその下にある頂点へ対角線を引く

### 走査線状態

- 走査線と交差する辺の集合
- これらの辺の右側に多角形がある
- 付随の Helper の頂点と共に左から右へ蓄えられる

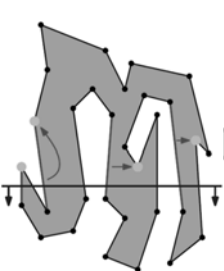
### 辺のヘルパー

- 辺 $e$ のヘルパー頂点 $v$ の特徴
  - 走査線より上に位置する一番低い頂点 $v$
  - $e$ と $v$ の間を連結する水平線分は多角形内にある
  - 唯一・固定ではなく、走査に伴い変化する



### データ構造

- 走査線状態
  - すべての辺(右に多角形がある)とそのHelperを左から右へ2分探索木の外点に蓄える
- イベント
  - 頂点
  - $y$ 座標で並び替える
  - リスト、キュー、配列に保存



### アルゴリズム

MakeMonotone( $P$ )

- 入力: 2重連結辺リスト $D$ に蓄えられた単純多角形 $P$
- 出力:  $D$ に蓄えられる $P$ を単調多角形へ分割した結果

1.  $y$ 座標値をキーとして、プライオリティキュー $Q$ を構成する( $y$ 座標値が同様な2点について、 $x$ 座標値が小さい方が優先)
2. 走査線と交差する $P$ の辺とそのヘルパーを蓄える2分探索木 $T$ を初期化
3. while  $Q \neq \text{empty}$
4. do  $Q$ の一番上から順番に $v_i$ を取り出す
5.  $v_i$ の種類に応じて処理を行う

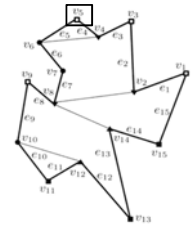
### 出発点の処理

HandleStartVertex( $v_i$ )

1.  $e_i$ を $T$ に挿入
2.  $e_i$ のヘルパー $helper(e_i)=v_i$ とする

$v_5$

- $e_5$ を $T$ に挿入
- $e_5$ のヘルパー「 $helper(e_5)$ 」= $v_5$




### 統合点の処理

HandleMergeVertex( $v_i$ )

1. if  $helper(e_{i-1})$ は統合点
2. then  $v_i$ と $helper(e_{i-1})$ を結ぶ対角線を $D$ に挿入
3.  $e_{i-1}$ を $T$ から削除
4.  $T$ を探索して、 $v_i$ のすぐ左の辺 $e_j$ を求める
5. if  $helper(e_j)$ は統合点
6. then  $v_i$ と $helper(e_j)$ を結ぶ対角線を $D$ に挿入
7.  $helper(e_j) \leftarrow v_i$

$v_4$

- $e_3$ のヘルパー頂点 $v_3$ は統合点ではないため、対角線を引かなく、走査線 $l$ は $e_3$ と $T$ が交差しなくなるので $T$ から $e_3$ を削除し、 $v_4$ のすぐ左の辺 $e_5$ を見つける。
- $e_5$ のヘルパー頂点 $v_5$ は統合点ではないため、対角線を引かなく、 $e_5$ のヘルパー頂点 $v_5$ を $v_4$ に変更。



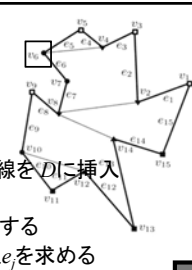
### 普通点の処理

HandleRegularVertex( $v_i$ )

1. if  $P$ の内部が $v_i$ の右にある
2. then if  $helper(e_{i-1})$ は統合点
3. then  $v_i$ と $helper(e_{i-1})$ を結ぶ対角線を $D$ に挿入
4.  $e_{i-1}$ を $T$ から削除
5.  $e_i$ を $T$ に挿入し、 $helper(e_i)=v_i$ にする
6. else  $T$ を探索して、 $v_i$ のすぐ左の辺 $e_j$ を求める
7. if  $helper(e_j)$ は統合点
8. then  $v_i$ と $helper(e_j)$ を結ぶ対角線を $D$ に挿入
9.  $helper(e_j) \leftarrow v_i$

$v_6$

- $helper(e_5)=v_4$ が統合点なので、 $v_6$ から $v_4$ へ対角線を引く
- 走査線 $l$ は $e_5$ と交差しなくなり $e_6$ と交差するようになるため、 $e_5$ を $T$ から削除し、代わりに $e_6$ を挿入
- $helper(e_6)=v_6$



### 分離点の処理

HandleSplitVertex( $v_i$ )

1.  $T$ を探索して、 $v_i$ のすぐ左の辺 $e_j$ を求める
2.  $v_i$ と $helper(e_j)$ を結ぶ対角線を $D_i$ に挿入
3.  $helper(e_j) \leftarrow v_i$
4.  $e_j$ を $T$ に挿入し、 $helper(e_j)=v_i$ とする

**V14**

- すぐ左の辺 $e_9$ の $helper(e_9)=v_8$ と $v_{14}$ を結ぶ対角線を加える
- $e_9$ のヘルパー頂点を $v_8$ から $v_{14}$ に変更 $helper(e_9)=v_{14}$
- $e_{14}$ を $T$ に挿入し、そのヘルパー頂点 $helper(e_{14})=v_{14}$ とする

### 最終点の処理

HandleEndVertex( $v_i$ )

1. if  $helper(e_{i-1})=$ 統合点
2. then  $v_i$ と $helper(e_{i-1})$ を結ぶ対角線を $D_i$ に挿入
3.  $e_{i-1}$ を $T$ から削除

**V15**

- $helper(e_{14})=v_{14}$  ≠ 統合点なので、対角線を挿入する必要はない
- $e_{14}$ を $T$ から削除

### 実行例

$v_5$   $v_3$   $v_4$   $v_6$   $v_7$   $v_1$   $v_9$   $v_2$   $v_8$   $v_{14}$   $v_{10}$   $v_{15}$   $v_{12}$   $v_{11}$   $v_{13}$

$l$   $T$

$e_5$   $helper$   $v_5$

- HANDLESTARTVERTEX( $v_5$ )  
 $e_5$ を $T$ に挿入し、 $e_5$ のヘルパーを $v_5$ とする。

### 実行例

$v_5$   $v_3$   $v_4$   $v_6$   $v_7$   $v_1$   $v_9$   $v_2$   $v_8$   $v_{14}$   $v_{10}$   $v_{15}$   $v_{12}$   $v_{11}$   $v_{13}$

$l$   $T$

$e_3$   $helper$   $v_5 \rightarrow v_4$   $v_3$

- HANDLEMERGEVERTEX( $v_4$ )  
 $e_5$ のヘルパー頂点も $e_3$ のヘルパー頂点も統合点ではないため、対角線を引かない。  
走査線 $l$ は $e_3$ と交差しなくなるため $T$ から削除し、 $e_5$ のヘルパー頂点を $v_5$ から $v_4$ に変更。

### 実行例

$v_5$   $v_3$   $v_4$   $v_6$   $v_7$   $v_1$   $v_9$   $v_2$   $v_8$   $v_{14}$   $v_{10}$   $v_{15}$   $v_{12}$   $v_{11}$   $v_{13}$

$l$   $T$

$e_6$   $helper$   $v_4 \rightarrow v_6$   $e_5$   $e_6$

- HANDLEREGULARVERTEX( $v_6$ )  
 $e_5$ のヘルパー頂点 $v_4$ が統合点なので、 $v_6$ から $v_4$ へ対角線を引く。  
走査線 $l$ は $e_5$ と交差しなくなり $e_6$ と交差するようになるので、 $e_5$ を $T$ から削除し、代わりに $e_6$ を挿入。  
 $helper(e_6)=v_4 \rightarrow v_6$

### 実行例

$v_5$   $v_3$   $v_4$   $v_6$   $v_7$   $v_1$   $v_9$   $v_2$   $v_8$   $v_{14}$   $v_{10}$   $v_{15}$   $v_{12}$   $v_{11}$   $v_{13}$

$l$   $T$

$e_8$   $helper$   $v_7 \rightarrow v_8$   $e_7$   $v_2$

- HANDLEMERGEVERTEX( $v_8$ )  
 $e_7$ のヘルパー頂点 $v_2$ が統合点なので、 $v_8$ から $v_2$ へ対角線を引く。  
走査線 $l$ は $e_7$ と交差しなくなるので $T$ から $e_7$ を削除し、 $e_9$ のヘルパー頂点を $v_9$ から $v_8$ に変更。

### 実行例

■ **HANDLESPPLITVERTEX( $v_{14}$ )**  
 $v_{14}$ と $v_8$  ( $e_9$ のヘルパー頂点)を結ぶ対角線を引く。  
 $e_9$ のヘルパー頂点を $v_8$ から $v_{14}$ に変更。  
 $e_{14}$ を $T$ に挿入し、そのヘルパー頂点を $v_{14}$ とする。

### 実行例

■ **HANDLEENDVERTEX( $v_{15}$ )**  
 $e_{14}$ のヘルパー頂点 $v_{14}$ は統合点ではないため、対角線を引かない。  
 $e_{14}$ を $T$ から削除。

### 実行例

■ **HANDLESPPLITVERTEX( $v_{12}$ )**  
 $v_{12}$ と $v_{10}$  ( $e_{10}$ のヘルパー)を結ぶ対角線を引く。  
 $e_{10}$ のヘルパーを $v_{10}$ から $v_{12}$ に変更。  
 $e_{12}$ を $T$ に挿入し、そのヘルパーを $v_{12}$ とする。

### 計算量

- プライオリティキュー $Q$ の構成 $\rightarrow O(n)$ 時間
- 2分探索木 $T$ の初期化 $\rightarrow O(1)$ 時間
- 1つ頂点に到達したときに行われる処理
  - $Q$ に関する操作(1回) $\rightarrow O(\log n)$ 時間
  - $T$ に関する質問(高々1回、挿入、削除(1回)) $\rightarrow O(\log n)$ 時間
  - $D$ への対角線の挿入(高々2本) $\rightarrow O(1)$ 時間
- 1つイベントに対して $O(\log n)$ 時間
- $n$ 個の頂点に対して処理を行うので、全体での実行時間は $O(n \log n)$

### 単調多角形 $\rightarrow$ 三角形

基本的な考え方

- $y$ 座標値の降順で一つずつ頂点を処理する
- 最高頂点から最低頂点まで、両側の境界をたどり、徐々に降りて行きながら、可能であれば、対角線を加え、三角形分割を行う

### 1つ頂点についての処理

- 準備=スタック $S$ 
  - $\Rightarrow$  既に出会ったが、また対角線を引ける頂点。
  - $\Rightarrow$  まだ対角線が引かれていない頂点。
- 処理=その頂点から $S$ 内にある頂点に向けて最大限に多数の対角線を引く
- $v_6$ の処理 $\Rightarrow S$ 内 $v_1 \sim v_5$

### ■ 処理頂点 $v_j$ と相手頂点の位置関係

- $v_j$  が  $S$  中の相手頂点と異なる側
- $v_j$  が  $S$  中の相手頂点と同じ側(左)
- $v_j$  が  $S$  中の相手頂点と同じ側(右)

### ■ 対角線の引き方①異なる側

- $v_j$  から  $S$  中にあるすべての頂点(最高の頂点「 $e$ の上端点」除外)へ対角線を引く
- これらの頂点を  $S$  から全部取り出す
- $v_j$  と最低の頂点を  $S$  に戻す

### ■ 対角線の引き方②同じ側(左)

- $v_j$  から  $S$  中にあるすべての頂点へ対角線を引けない可能性がある
- 最低頂点はすでに  $v_j$  と連結しているため、最低頂点の上にある複数の頂点を一つずつ取り出す
- 引ける頂点まで対角線を引く
- $v_j$  と最後の対角頂点を  $S$  に戻す

### ■ 対角線の引き方③同じ側(右)

- $v_j$  から  $S$  中にあるすべての頂点へ対角線を引けないため、取り出された頂点を再び  $S$  に戻す
- $v_j$  を  $S$  に戻す

### ■ アルゴリズム

TriangulateMonotonePolygon( $P$ )

- 入力:  $D$  に蓄えられた  $y$  単調な多角形  $P$
- 出力:  $D$  に蓄えられた  $P$  の三角形分割

- $P$  の左右側のチェーン上の頂点列を一つの系列に統合し、 $y$  座標値の降順に並び替える  $u_1, \dots, u_n$
- $u_1$  と  $u_2$  をスタック  $S$  に蓄える
- for  $j=3$  to  $n-1$
- if  $u_j$  と  $S$  の一番上の頂点異なる側チェーン上にある
- then  $S$  からすべての頂点を取り出す
- $u_j$  と取り出されたそれぞれの頂点を結ぶ対角線を  $D$  に挿入(最後の頂点を除外)
- $u_{j-1}$  と  $u_j$  を  $S$  に入れる

### ■ アルゴリズム 続き

- else  $u_j$  と  $S$  の一番上の頂点と同じ側チェーン上にある
- $S$  から1つ頂点を取り出す
- $u_j$  からの対角線が  $P$  の内部にある限り、 $S$  から他の頂点を取り出す。これらの対角線を  $D$  に挿入。取り出された最後の頂点を  $S$  に入れる
- $u_j$  を  $S$  に入れる
- 最初と最後の頂点を除いて、 $u_n$  から  $S$  上の全ての頂点への対角線を加える



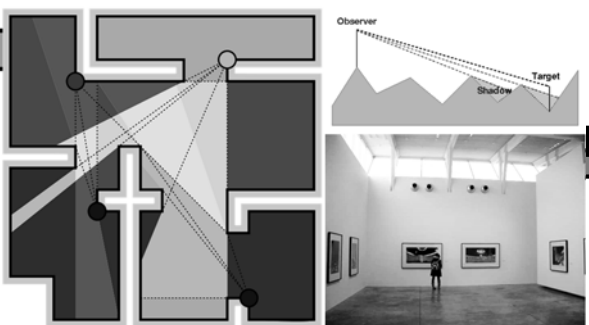
### 計算量

- $u_1$ と $u_2$ をスタック $S$ にプッシュ→ $O(1)$ 時間
- $u_3 \sim u_{n-1}$ の処理( $n-3$ 回の繰り返し処理)
  - スタック $S$ へのプッシュの回数  
→1つ頂点の処理につき高々2回
  - スタック $S$ からのポップの回数  
→プッシュの回数以下  
⇒ $O(n)$ 時間
- $n$ 頂点を持つ $y$ 単調な多角形は、線形時間で三角形分割できる。

### 応用

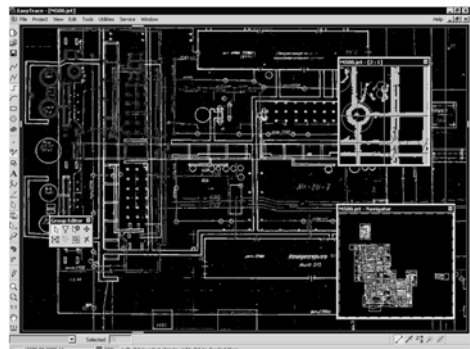
- 可視性と最短経路問題
  - 美術館監視、警備員巡回、要塞防衛、刑務所警備
  - ロボット最適経路設計
- 有限要素法セル分割
- VLSI設計
- 画像処理
- コンピュータグラフィクス

### 可視性問題



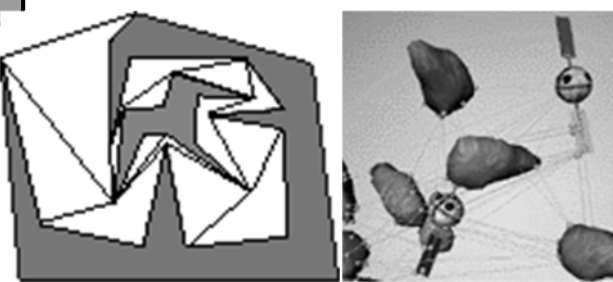
The diagram shows a 2D environment with several obstacles. An 'Observer' is located at a point, and a 'Target' is at another point. A visibility graph is constructed by connecting the observer to visible vertices and those vertices to the target. A path is highlighted from the observer to the target. An inset shows a 3D scene with a person and a target, with a 'Shadow' cast by an object, illustrating visibility constraints.

### 画像処理



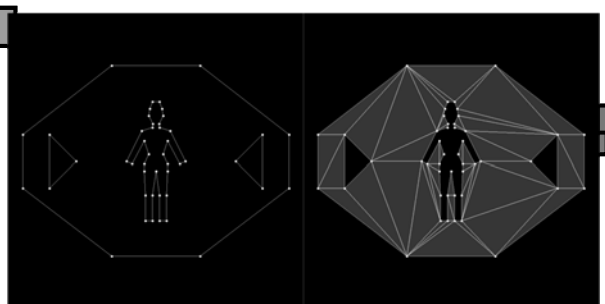
A screenshot of a computer graphics application displaying a complex circuit board layout. The board is filled with various components and traces, illustrating the application of computational geometry in image processing and layout design.

### 衝突検出



The diagram illustrates collision detection. On the left, a 3D model of a robot moving through a complex environment with obstacles. On the right, a 2D representation of the robot and obstacles with lines connecting them to show potential collision points or visibility lines.

### 内部空穴の特定



The diagram shows two views of a polygon with an internal hole. The left view shows the hole as a separate polygonal region within the main shape. The right view shows the hole filled with a triangulation, illustrating how internal holes are identified and processed in computational geometry.

