8 プログラム構成

8-1 Speechio.dot

Microsoft Word のアドイン形式。主に以下の目的で開発を行っている。

- Word 上からの関数の呼び出し（Word 上へのインターフェイスの実装）
- Word VBA の構文解析機能（Sentence オブジェクト）や画像処理機能（Shape オブジェクト）の利用

ツールバーとメニューから呼び出されるプロシージャは、下表のようになっている。

表 9　インターフェイス上から実行されるプロシージャ

アイコン	機能	ファイル名	プロシージャ
 （青）	ワンクリック（男性の声）	mdlSp1Click.bas	SpOneClickMale
 （赤）	ワンクリック（女性の声）	mdlSp1Click.bas	SpOneClickFemale
 （黄）	詳細設定	mdlSpAdvanced.bas	SpAdvanced
	シール印字	mdlSpSeal.bas	SpSealPrinting
	読みの確認	mdlSpVoice.bas	SpReading
	環境設定	mdlSpSetting.bas	SpSetting

8-2 SP_AUTH.dll

DLL 形式の音声コード作成エンジン。主に以下の目的で開発を行っている。

- 文字列処理の速度向上
- ダイアログのスクリーンリーダー（視覚障害者向けの PC 画面読上げソフトウェア）への対応（VB のフォームはコントロールによってスクリーンリーダーに未対応のものがあるため、C 言語＋Windows API にてダイアログを作成することで対応している。）
- 東芝音声合成ライブラリの呼び出し（スタティックリンクライブラリとして組み込み）
- 音声コード読取機器ソースの移植（音声コード読取機器では全て C 言語ベースで開発されている）

「Speechio.dot」の内部から呼び出すことを前提としているため、VB から呼び出し可能な形式（__stdcall 呼び出し規約）で開発を行っている。

8-3 Speechio.dll

DLL 形式の SP コード作成エンジン。API 関数などの詳細は、「付録 1 SP Code 作成ソフトウェア 開発者向け仕様書」を参照のこと。また、「Speechio.dot」内では、「mdlSpCode.bas」にて「Speechio.dll」の API 関数が宣言されている。

9 設定ファイルの構成

設定ファイル（Speechio.ini）は、基本的に環境設定ダイログ（「7-4 環境設定」参照）から変更するが、設定ファイルをエディタなどで直接変更することも可能である。設定ファイル内に記述される内容は、表 9 のような構成になっている。

表 9　設定ファイルの構成

セクション名		キー名		値	
項目	意味	項目	意味	値	意味
EncodeSetting	エンコード設定	DataType	データタイプ	T	日本語テキスト
				C	コントロールカード
		CellType	コードサイズ	s	XS サイズ
				S	S サイズ
				m	M サイズ
				M	L サイズ
				〈空〉	指定のない場合は自動
		RecoverLevel	誤り訂正	S	強
				N	中
				P	弱
SpProperty	音声コード作成の設定	Position	コードの位置	0	右下
				1	左下
				2	偶数ページは右下
				3	偶数ページは左下
		PageBreak	自動改ページ	-1	有り
				0	無し
		EncodeRange	変換対象範囲	0	全文
				1	選択範囲、未選択は全文
				2	選択範囲のみ
		PageInfo	ページ情報	整数	0（なし） 1（ページ番号） 2（文書名） 4（コメント） の合計値
		PageInfoGender	ページ情報の声質	0	男声
				1	女声
				2	自動（ページ先頭と同じ）
		Comment	コメント	文字列	

		SaveText	テキスト出力	-1	有り
				0	無し
		SaveBMP	BMP 画像出力	-1	有り
				0	無し
		SaveDir	保存先	Select	作成の都度指定
				OpusApp	Word 文書と同じ
				〈パス〉	常にこのフォルダへ出力
Others	その他	LatestDir	前回の保存先	〈パス〉	前回の〈パス〉を記録
SpVoice	声質設定	Gender	声質	0	男
				1	女
		Speed	速さ	整数	-20～20
		Pitch	高さ	整数	-100～100
		Volume	大きさ	整数	-8～8
SealPrinting	シール印字設定	PaperWidth	用紙の幅	数値	
		PaperHeight	用紙の高さ	数値	
		Orientation	用紙の方向	0	縦
				1	横
		TopMargin	上余白	数値	
		BottomMargin	下余白	数値	
		LeftMargin	左余白	数値	
		RightMargin	右余白	数値	
		Hspace	シールの横の間隔	数値	
		Vspace	シールの縦の間隔	数値	
		Col	列数	整数	
		Row	行数	整数	
#	コメント行				

設定ファイル内に記述されている文法は、Windows で使用される一般的な ini ファイルと同じで、Windows API の GetProfileString 関数や WriteProfileString 関数などを使用して読み書きを行う。

10 Word アドインの開発について

ここでは、Word に組み込んだエンコーダープログラムの環境、起動、実行について説明する。

10-1-1 開発時のファイルの位置

Microsoft Word VBA にて開発を行う場合、フォームやモジュールなどのデータは、全て既定のアドインファイルである「Normal.dot」へ保存する。今回の開発環境である Word 2000/Windows 2000 の場合は、以下の場所に「Normal.dot」が存在する。

C:¥Documents and Settings¥〈ユーザ名〉¥Application Data¥Microsoft¥Templates

既存の「Normal.dot」は削除しても構わず、存在しない場合は Word の起動時に自動的に作成される。また、上記の場所にアドインを配置しても「Normal.dot」以外のファイル名としておけば、Word の起動時にロードされることはない。

既製のアドイン（Speechio.dot など）を変更したい場合は、上記位置に「Normal.dot」としてコピーすれば編集可能となる。

10-1-2 エディタの起動

Microsoft Word 上にて「ツール」メニューの「マクロ」から、「Visual Basic Editor」を選択する。画面左側に表示される「プロジェクトエクスプローラ」から、「Normal」を選択して開発を行う。Word のバージョンによって VB のバージョンも異なり、使用できない関数などが存在するため注意が必要である。

10-1-3 マクロの実行

作成したマクロは、Word 上のツールバーやメニューから実行可能である。新しくツールバーやメニューを作成する場合は、「ツール」メニューの「ユーザー設定」を選択して表示される、「ユーザー設定」ダイログを使用する。

ツールバーを作成するには、「ユーザー設定」ダイログの「ツールバー」タブを選択して、「新規作成」ボタンを押し、表示されるダイログでツールバー名と保存先を指定する。また、メニューを作成するには、「ユーザー設定」ダイログの「コマンド」タブを選択して、「分類」一覧の一番下にある「新しいメニュー」を選択する。「コマンド」一覧に「新しいメニュー」が表示されるので、これを既存のメニューバーやツールバー上にドラッグ＆ドロップする。

次に、作成したツールバーやメニュー内にマクロを登録する。「ユーザ設定」ダイログの「コマンド」タブを選択し、「分類」一覧からマクロを選択すると、「コマンド」に利用可能なマクロの一覧が表示される。このマクロをツールバーやメニューへドラッグ＆ドロップして、目的の順番となるよう配置する。

また、「ユーザ設定」ダイアログを表示したまま、登録したマクロを右クリックすると、名前やボタンイメージなどが変更可能となる。

10-1-4 マクロの設定

作成したマクロを設定する際には、Word 上で「スタートアップ」として設定されているフォルダにコピーする必要がある。Word 上からは、以下の手順によって「スタートアップ」に指定されているフォルダを確認でき、このフォルダ内にマクロを配置する。

① 「ツール」メニューから「オプション」を選択
② 「既定のフォルダ」タブをクリックして表示
③ ファイルの種類一覧で「スタートアップ」の既定の参照先を確認

「スタートアップ」フォルダは、使用する Word や Windows のバージョンによってパスが異なるため、注意が必要である。デフォルトのパスは一般的に以下のようになっている。

Word 97/98

C:¥Program Files¥Microsoft Office¥Office¥STARTUP

Word 2000/2002

・Windows 9x の場合

C:¥Windows¥Application Data¥Microsoft¥Word¥STARTUP

・Windows NT 4.0 の場合

C:¥Windows¥Profile¥<ユーザ名>¥Application Data¥Microsoft¥Word¥STARTUP

・Windows 2000/XP の場合

C:¥Documents and Settings¥<ユーザ名>¥Application Data¥Microsoft¥Word¥STARTUP

また、アドインのファイル名は任意であるが、拡張子を必ず「.dot」とする必要がある。

10-1-5 アドインの保護

本ソフトウェア中のアドイン「Speechio.dot」では、内容を表示するためにパスワード入力が必要となるよう保護を行っている。以下の手順で、内容の表示とパスワードの変更が可能である。

① 「Speechio.dot」を既定のアドイン「Normal.dot」としてコピーする。
② Word を起動し、「ツール」メニューの「マクロ」から「Visual Basic Editor」を選択する。
③ 「プロジェクトエクスプローラ」から「SP_CODE_MAKER (Normal)」を選択する。
④ 表示されるパスワード入力のダイアログで、「r2d2mateC」と入力すると、プロジェクトが編集可能となる。パスワードを変更するためには、「Visual Basic Editor」上の「ツール」メニューの「SP_CODE_MAKER のプロパティ」を選択し、表示されるプロジェクトプロパティダイアログにある「保護」タブにて変更する。

仕様説明

以下、断りがない限り、

文字コード：シフト JIS コード（半角カナ、半角 ASCII 文字を含む）

SP 制御コード：音声コード読み取り機器からの出力用制御コード

SP format：SP 制御コード ＋ テキスト

とする。

1-1 音声出力

音声出力のため、音声エンジンに送信する音声データへ変換する。

1-1-1 処理の流れ

音声データ変換モジュールの処理流れを図 1-1-1 に示す。

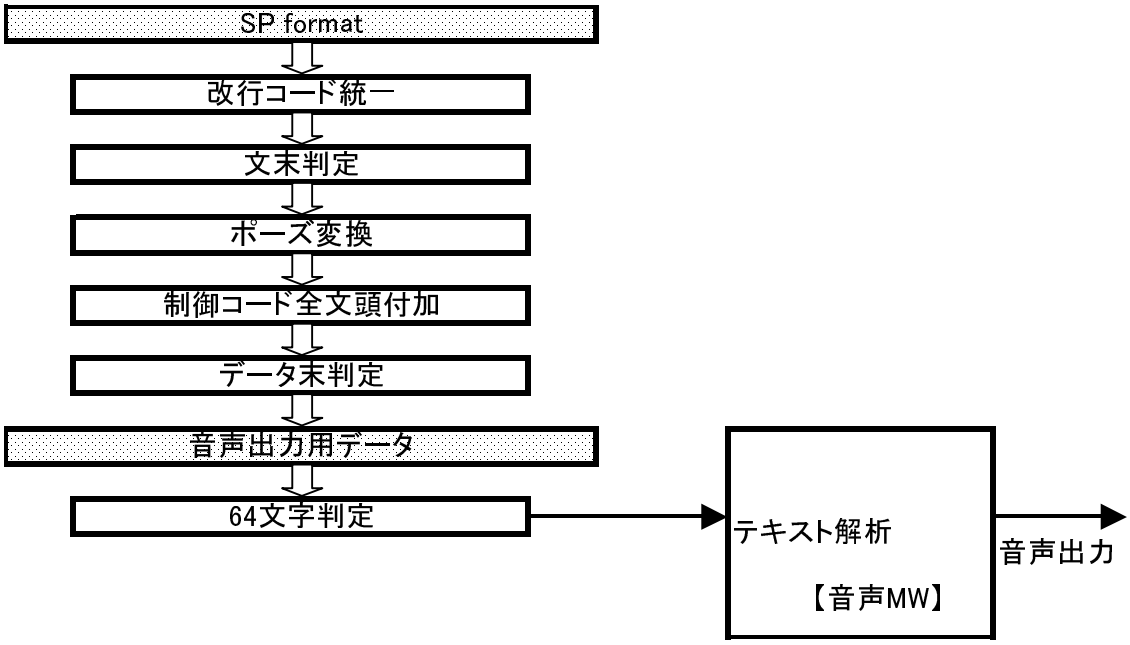


図 1-1-1 音声データ変換モジュールの処理流れ

1-1-2 音声出力用データ作成

1-1-2-1 改行コード統一

データ作成をした OS、ソフトウェアによって改行コードに違いがある。

そこで改行コードを CR(0x0d)+LF(0x0a)に統一する。

改行コード：CR+LF

1-1-2-2 文末判定

文単位で音声出力を行うため、文末を判定し NULL(0x00)を挿入し、文単位に区切る。

文末文字を表 1-1-1 に示す。

表 1-1-1 文末文字

。）	。)	？）	?)	！）	!)	。
。」	。」	？」	?」	！」	!」	。
。］	。］	？］	?］	！］	!］	？
。｝	。｝	？｝	?}	！｝	!}	？
。〉	。〉	？〉	?〉	！〉	!〉	！
。》	。》	？》	?》	！》	!》	！
。」	。」	？」	?」	！」	!」	CR+LF
。』	。』	？』	?』	！』	!』	
。】	。】	？】	?】	！】	!】	
。）	。）	？）	?)	！）	!)	
。］	。］	？］	?］	！］	!］	
。｝	。｝	？｝	?}	！｝	!}	
。」	。」	？」	?」	！」	!」	
。”	。″	？”	?″	！”	!″	
。’	。'	？’	?'	！’	!'	

ここで、文末文字は連続しないものとする。

(1) 直前に NULL が挿入されている場合は、文末文字を削除し、NULL は挿入しない。

(2) 直前に NULL+スペースが挿入されている場合は、スペース、文末文字を削除し、NULL は挿入しない。

1-1-2-3 ポーズ変換

音声出力の際、改行、TAB で一定の間隔を空けるため、間隔の空く記号に変換する。

(1) 改行

改行[CR+LF]は、"。"(0x8142)に変換し、NULL を挿入する。

(2) TAB

TAB(0x09)は、"、"(0x8141)に変換する。

(2-1) 直前に NULL が挿入されている場合は、[TAB]を削除する。

(2-2) 直前が"、"の場合は、[TAB]を削除する。

1-1-2-4 機種依存文字

文章中に機種依存文字が入っていると、音声エンジンの仕様により、予期せぬ動作を示す場合がある。

そこで、機種依存文字（0x84xx～0x87xx）を半角文字に変換する。

0x84xx～0x87xx までの文字で表 1-1-2 にない文字はスペース(0x20)に変換する。

表 1-1-2 機種依存文字変換表

全角	半角	全角	半角	全角	半角	全角	半角
①	(1)	⑪	(11)	Ⅰ	I	㎜	mm
②	(2)	⑫	(12)	Ⅱ	II	㎝	cm
③	(3)	⑬	(13)	Ⅲ	III	㎞	km
④	(4)	⑭	(14)	Ⅳ	IV	㎎	mg
⑤	(5)	⑮	(15)	Ⅴ	V	㎏	kg
⑥	(6)	⑯	(16)	Ⅵ	VI	㏄	cc
⑦	(7)	⑰	(17)	Ⅶ	VII		
⑧	(8)	⑱	(18)	Ⅷ	VIII		
⑨	(9)	⑲	(19)	Ⅸ	IX		
⑩	(10)	⑳	(20)	Ⅹ	X		

1-1-2-5 読み仮名指定

SP format には、音声出力時の読み仮名の情報が付加されている場合がある。

そこで、単語データを削除する。

例えば、(表記:ﾋｮｳｷ)と読み仮名の指定がされている場合、”ヒョウキ”と発音する。

音声出力では読み仮名部分“ﾋｮｳｷ”のみ必要なので、単語データを削除し、読み仮名部分を取り出す。

1-1-2-6 SP 制御コード全文頭付加

文章の先頭に、前文までの SP 制御コードの設定を更新し付加する。

これは、SP 制御コードにより設定される文章を飛ばしても、正しい設定で再生する為に行う。

例）声質設定の制御コードを[男声]：男性の声、[女声]：女性の声とする。

文章 1：[男声]僕は男です。

文章 2：女の子が好きです。

文章 3：[女声]私は女です。

この時、文章 1-2-3 と再生された時は問題ないが、文章 3 まで再生後に文章 2 に戻った時に、文章 3 を再生した時点で音声エンジンは女性の声に設定されているので、文章 2 は女性の声で再生されることになる。

これを全文章の頭に前文までの SP 制御コードを付加することにより、

文章 1：[男声]僕は男です。

文章 2：[男声]女の子が好きです。

文章 3：[女声]私は女です。

文章 3-2 と再生されても、正しい設定で再生できる。

1-1-2-7 データ末判定

音声データ末に NULL(0x00),NULL(0x00)を挿入し、データ末を区切る。

1-1-3 データ出力

音声出力用データを音声エンジンに介してスピーカー（PC）から出力する。

音声エンジンへのデータ送信は OS に準拠された仕様で送られる。

1-2 テキスト出力

テキスト出力のため、音声出力用制御コード等、テキスト以外の情報を削除する。

1-2-1 処理の流れ

テキストデータ変換モジュールの処理流れを図 1-3-1 に示す。

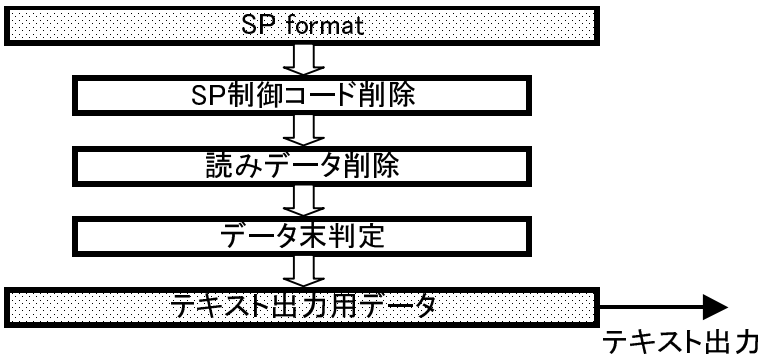


図 1-3-1 テキストデータ変換モジュールの処理流れ

1-2-2 テキスト出力用データ作成

1-2-2-1 SP 制御コード削除

SP format にはテキスト以外の情報の SP 制御コードが付加されている。

そこで CR[0x0d]、LF[0x0a]、TAB[0x09]、NULL[0x00]以外の制御コードを削除する。

1-2-2-2 読み仮名指定

SP format には、音声出力時の読み仮名の情報が付加されている場合がある。

そこで読み仮名データを削除する。

例えば、(表記:ﾋｮｳｷ)と読み仮名の指定がされている場合、”ヒョウキ”と発音する。

テキスト出力では単語部分“表記”のみ必要なので、読み仮名データを削除し、単語部分を取り出す。

1-2-2-3 データ末判定

テキストデータ末に NULL(0x00),NULL(0x00)を挿入し、データ末を区切る。

1-2-3 データ出力

テキスト出力用データを画面表示へと出力する。

出力されたデータを文字サイズ変更、表示カラー変更設定により文字表示変更させる。

1-3 関数一覧

uchar：typedef unsigned char

SP_PRINTER_CONTROL 構造体：SP player の項を参照。

BRLCODE 構造体：

```
typedef struct {
    int index;          /* 英数記号もしくは、かな漢字の開始地点 */
    uchar* code;        /* 中間言語を取得したバッファのアドレス */
    int length;         /* 英数記号もしくは、かな漢字の長さ */
    int flg;            /* 英数記号、かな漢字判別フラグ */
} BRLCODE;
#define   FLG_CLEAR      0       /* デフォルトのフラグ */
#define   FLG_ALNUM      1       /* 英数記号のフラグ */
#define   FLG_NOTALNUM   2       /* かな漢字のフラグ */
```

1-3-1 音声出力

関数名：int SPtoVoice(uchar* voice_data,uchar* sp_data,int sp_size)

引き数：(入力) sp_data:デコード直後の SP format

sp_size:sp_data のサイズ

(出力) voice_data:受け取りバッファ

戻り値：voice_data のサイズ

機　能：全文頭に制御コードをつけた音声エンジン送信用データを作成。

注　意：受け取りバッファサイズは入力サイズの 2 倍以上確保。

1-3-2 テキスト出力（画面表示）

関数名：int SPtoText(uchar* text_data,uchar* sp_data,int sp_size)
引き数：(入力) sp_data:デコード直後の SP format
sp_size:sp_data のサイズ
(出力) text_data:受け取りバッファ
戻り値：text_data のサイズ

機　能：音声出力用制御コードを削除し、テキスト出力用データを作成。

2　デコードプログラム生成

【目的】
it 機上の SP コードデモアプリケーションにおける、スキャン画像からの SP コード抽出・デコードモジュールの設計仕様について述べるものである。

【概要】
SP コードデコーダでは、以下の処理を行う。

- スキャナ画像からの SP コード部分抽出
- 抽出した SP コードデータからのテキスト抽出

本モジュールは、SP コード読み取り機器　スピーチオのソースを Windows 上にポーティングすることにより実現している。
したがって、基本アルゴリズムはスピーチオ実機と同等のものである。

2-1 動作環境・開発環境

動作環境
本モジュールは、実装ハード（it 機）に依存する部分がないため、汎用 Windows 用 DLL として実装する。
したがって、本モジュールの動作環境は以下の通りとなる。
Windows 2000 Professional/Server 以降

開発環境
本モジュールは、以下の環境で開発する。
Microsoft Visual C++ V6.0 SP5

2-2　I/F 仕様

本モジュールは、以下の 2 つの関数をエクスポートする。

- Generator
  画像データからの SP コード取り出し
- Decoder
  SP コードデータからのテキスト抽出

2-2-1　画像データからの SP コード取り出し

形式

```
int Generator(unsigned char *img_data,
              int *row, int *col, unsigned char **bit_string)
```

引数

名前	I/O	説明
img_data	in	画像データ。1024x1024 ピクセルの 2 値データ
row	out	副走査方向の SP コードユニット数
col	out	主走査方向の SP コードユニット数
bit_string	out	SP コードデータビット文字列。”0”, “1”で構成される。

戻り値

0	成功
負値	失敗

概要

引数で指定された画像から、SP コード部分を抜き出す。
bit_string で返されるメモリは開放の必要なし。
スレッドセーフではないので注意が必要。

2-2-2 SP コードデータからのテキスト抽出

形式

int Generator(int row, int col, unsigned char *bit_string,
unsigned char **data)

引数

名前	I/O	説明
bit_string	in	SP コードデータビット文字列。”0”, “1”で構成される。
row	in	副走査方向の SP コードユニット数
col	in	主走査方向の SP コードユニット数
data	out	抽出テキスト

戻り値

0	成功
負値	失敗

概要

引数で指定された SP コードデータからテキストを抽出する。
data で返されるメモリは開放の必要なし。
スレッドセーフではないので注意が必要。

3 SP コード抽出処理詳細

SP コード抽出を行う上で、対象ハードの違いにより SP コード抽出処理は単純なポーティングでは動作しないことがわかっている。

そのため、元ソースを解析した結果を元に以下の通り、SP コード抽出処理のポーティングを実施した。

以降、元ソースの解析結果を記す。

3-1 ReadSymbol()関数

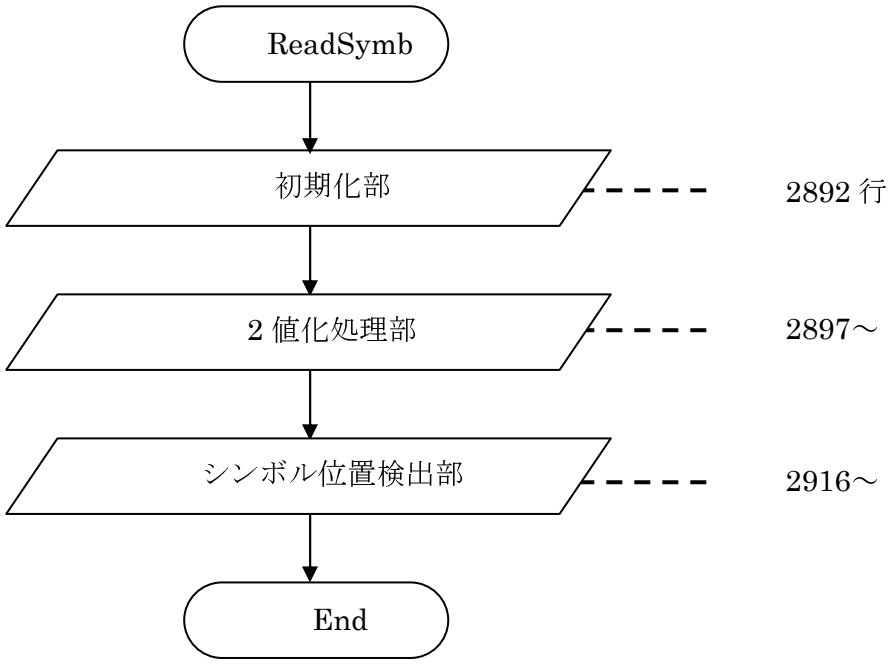
画像データから、2 次元コードビット列データを作成する。

【引数】

gray_buf : 8 ビットグレイスケール 画像データ（1024×1024 ピクセル）
xsize : シンボルの横のバイト数
ysize : シンボルの縦のバイト数
row : シンボルの横のセル数

col　　：シンボルの縦のセル数
data　　：エンコード用ビット列データを返すバッファ
【戻り値】
SUCCESS　成功
FAILURE　失敗

3-2　全体の流れ



3-3　各部詳細

以下にそれぞれの関数一覧を記す。

(1)初期化部

関数	定義位置	説明
InitSymbolReader	2613～2625 行	初期化する。

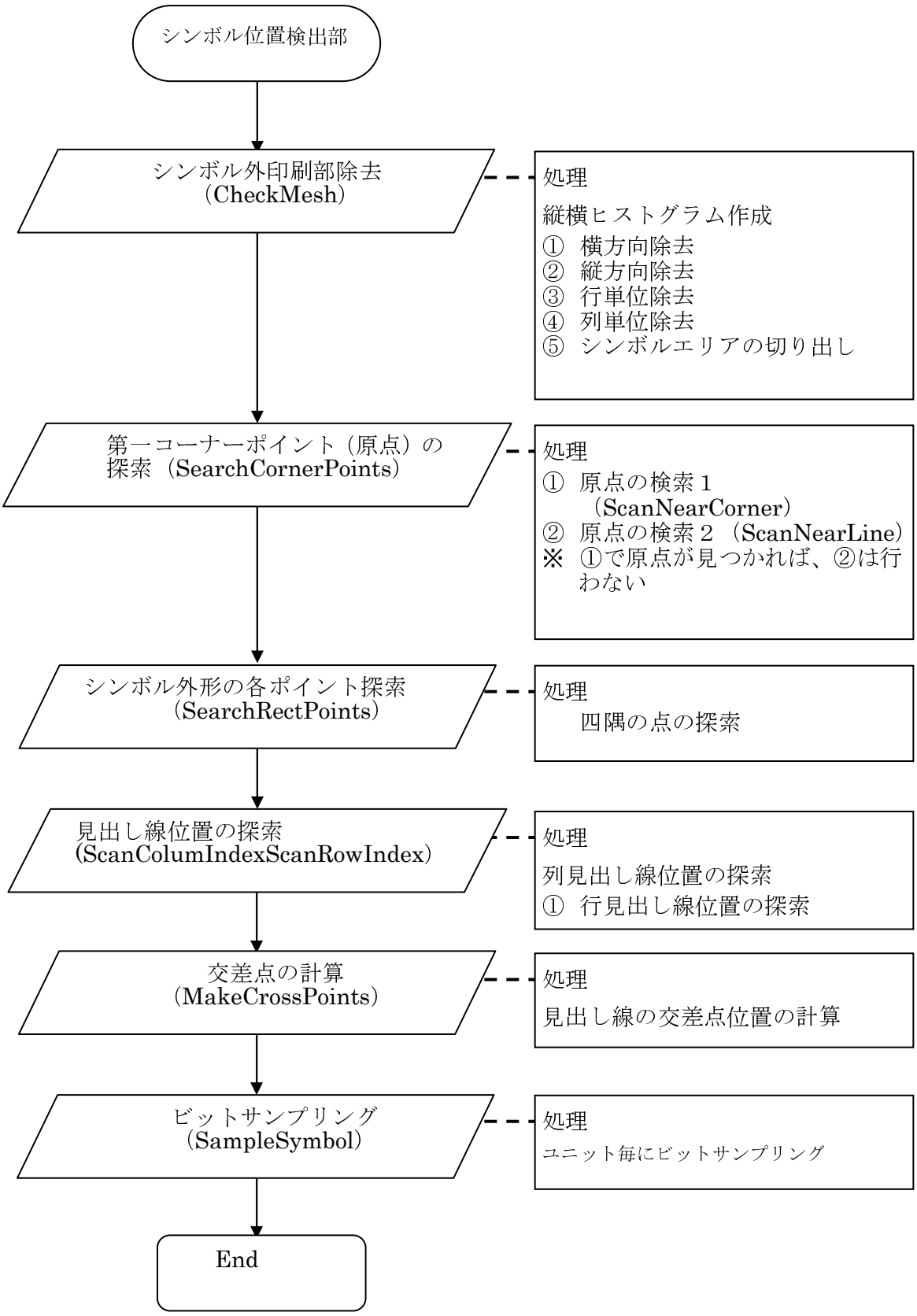
(2)2 値化処理部

関数関数	定義位置	説明
ModeBinarization	436～568 行	モード法により閾値を決定する。
SetupValidRect	423～434 行	エッジを強調する。
Binarization2	702～743 行	2 値化処理を行う。
KillBinaryNoise	816～866 行	ノイズを除去する。

(3)シンボル位置検出部

関数関数	定義位置	説明
CheckMesh	2629～2880 行	シンボル外印刷部を除去する。
SearchCornerPoints	2499～2523 行	第一コーナーポイントを探索する。
SearchRectPoints	1723～1909 行	シンボル外形の各ポイントを探索する。
ScanColumIndex	2448～2471 行	見出し線位置（列）を探索する。
ScanRowIndex	2473～2496 行	見出し線位置（行）を探索する。
MakeCrossPoints	1949～1970 行	見出し線が交差する点を計算する。
SampleSymbol	2547～2609 行	ビットをサンプリングする。

シンボル位置検出の流れ



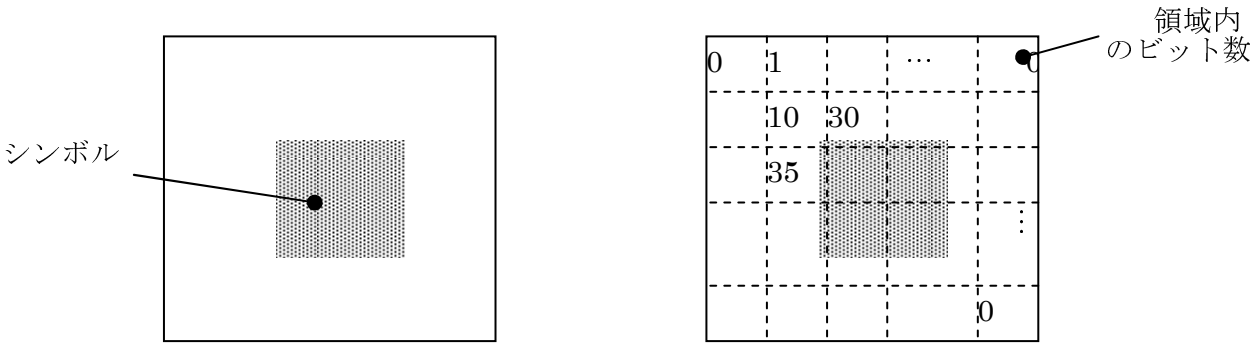
4　各処理の詳細

4-1 シンボル外印刷部除去

2 値画像データの水平垂直の分布状態を調査して、2 次元コード（シンボル）の存在する矩形エリアを特定する。

① 縦横ヒストグラムの作成

2 値画像データ（bin_buf）を 16×16 の領域に分割し、ビットの度数分布(HistMesh)を作成する。

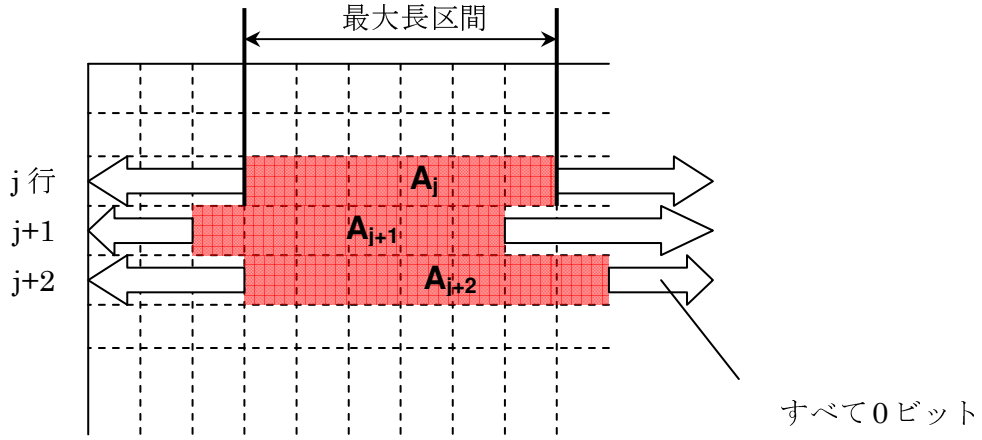


② 横方向除去

横方向に、ビット数が 2 以上連続して存在する最大長の区間を A とする。

HistMesh 各行で、区間 A を除く領域を全て 0 ビットとする。

0 ビットとした HistMesh 領域内の bin_buf も同様に 0 ビットとする。



HistMesh[16][16]

③　縦方向除去

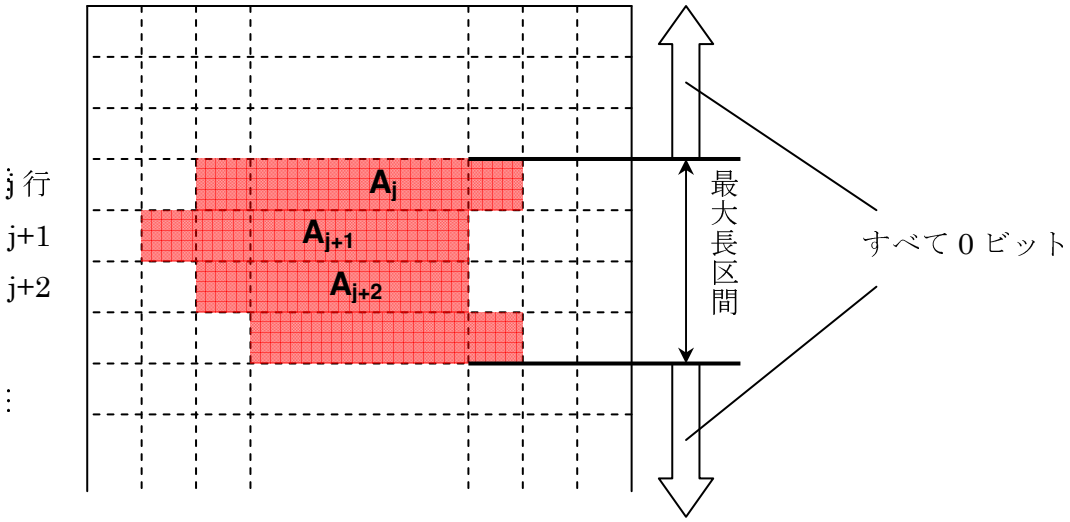
横方向除去と同様の手順で、縦方向に 0 ビットとする。

④　行単位除去

行単位で、ビットのある領域の最大長区間を探す。

最大長区間を除く領域を全て 0 ビットとする。

0 ビットとした HistMesh 領域内の bin_buf も同様に 0 ビットとする。

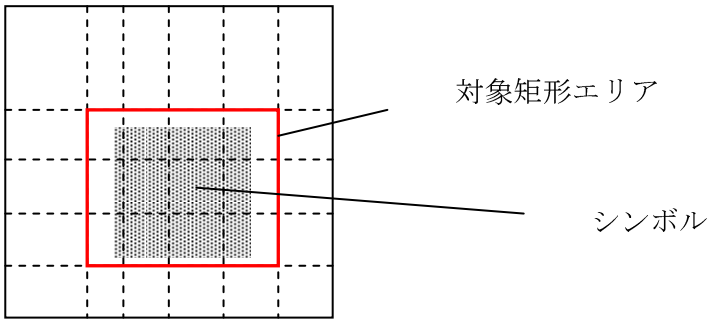


HistMesh[16][16]

⑤　列単位除去

行単位除去と同様の手順で、列単位に 0 ビットとする。

②～⑤の処理後、対象矩形エリア以外全て 0 ビットとなる。



HistMesh[16][16]

⑥　シンボルエリアの切り出し

対象矩形エリア内の、横方向（x）及び縦方向（y）の最大/最小位置を探索する。

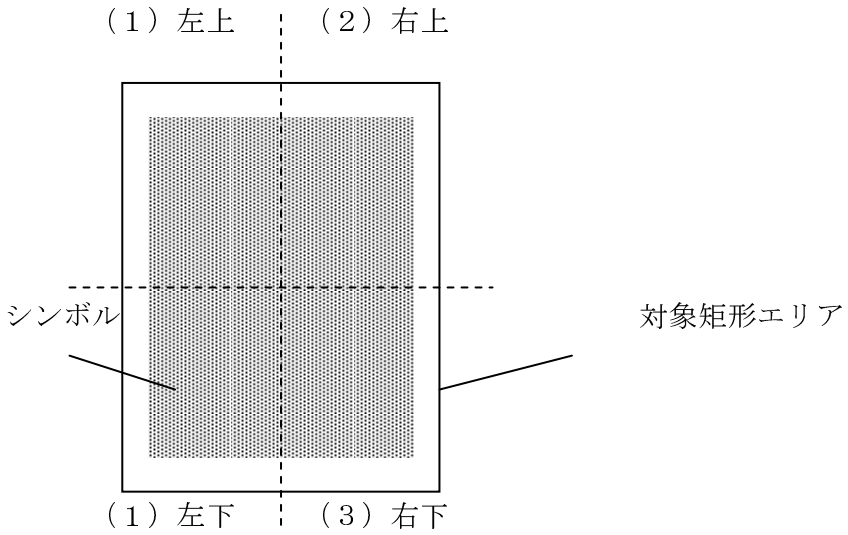
4-2　第一コーナーポイントの探索

コーナーポイント（P0、P1、P2、P3）を探索し、第一コーナーポイント（原点 Pt0）を検索する。

① 原点の検索１

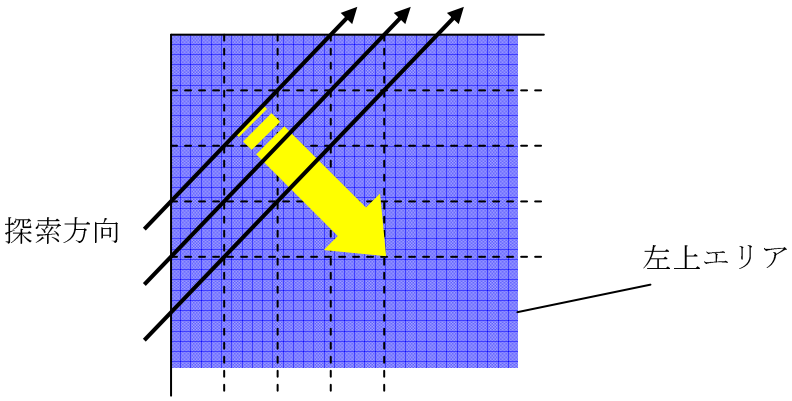
【探索エリアとコーナーポイントの探索】

対象矩形エリアから、左上コーナー⇒右上コーナー⇒右下コーナー⇒左下コーナーの順で、コーナーポイントを探索する。



（１） 左上エリア

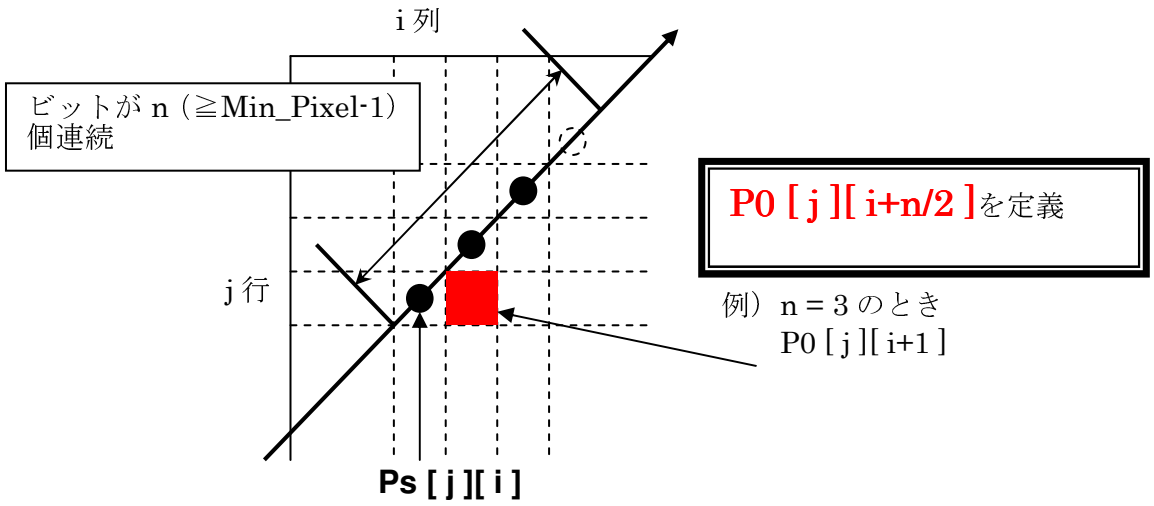
左上コーナーから対角方向に、下図のように斜めに探索する。



探索方向にビット数が（Min_Pixel -1）以上連続して存在する場合に、左上コーナーポイント（P0）を定義する。このとき、連続開始点を Ps [j][i]とする。

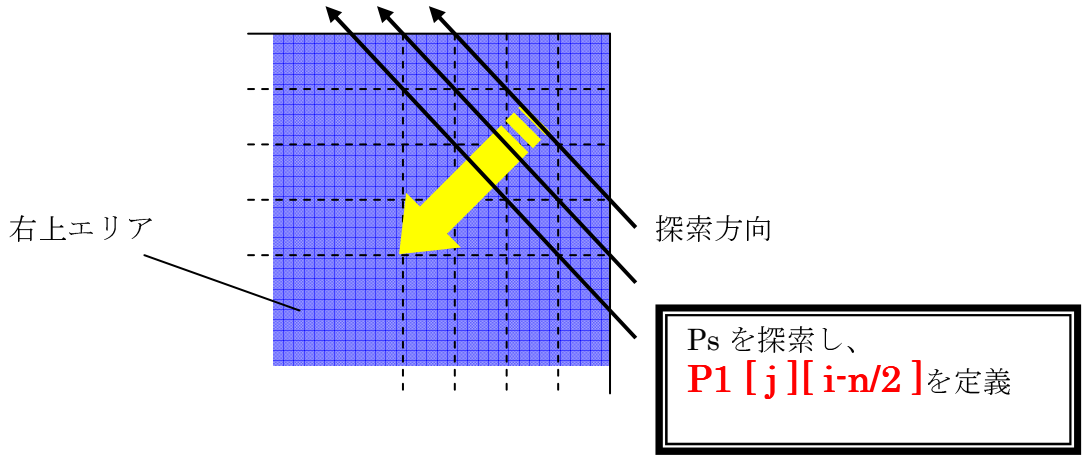
ここで、Min_Pixel は、１ドットあたりの最小ピクセル数を示す。

※本バージョンでは、Min_Pixel = 4 である。



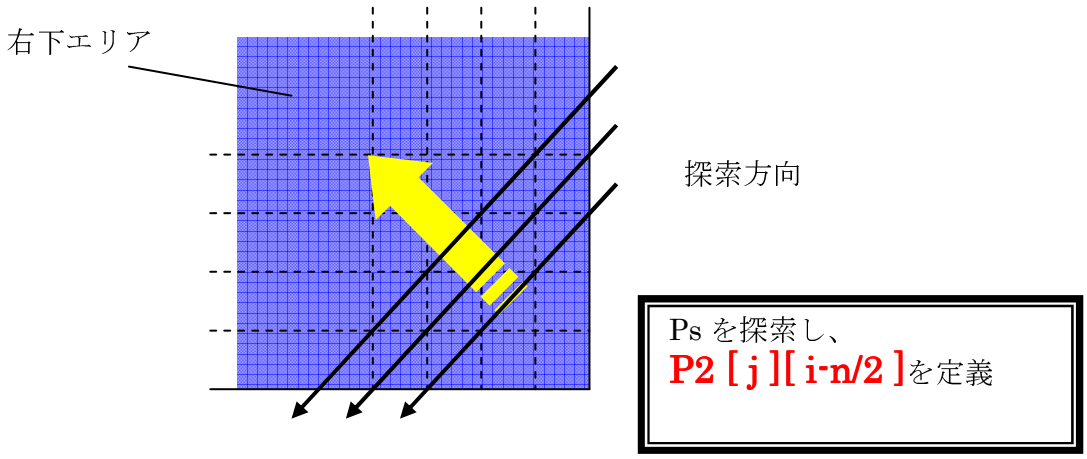
（2） 右上エリア

右上コーナーから対角に向かって、下図のように斜めに探索し、右上コーナーポイント（P1）を定義する。



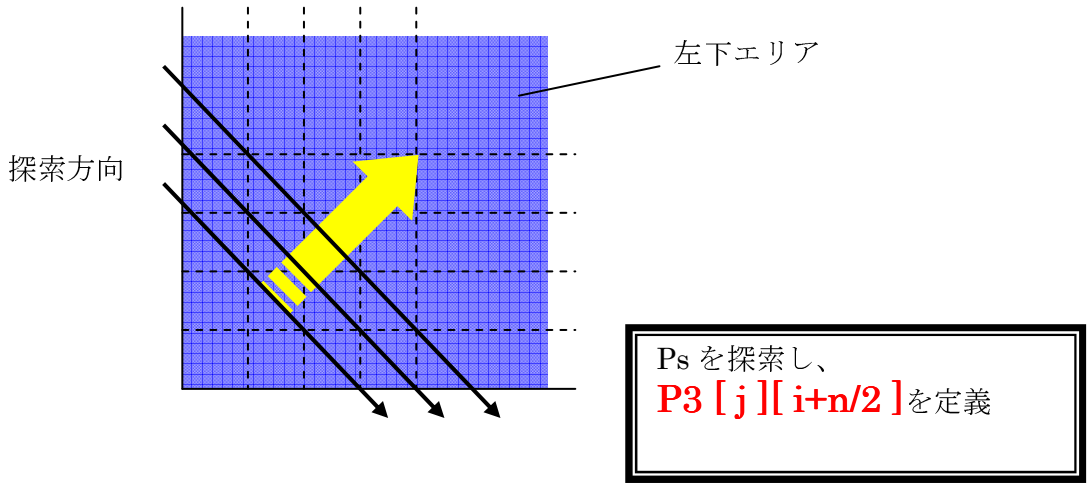
（3） 右下エリア

右下コーナーから対角に向かって、下図のように斜めに探索し、右下コーナーポイント（P2）を定義する。



（４） 左下エリア

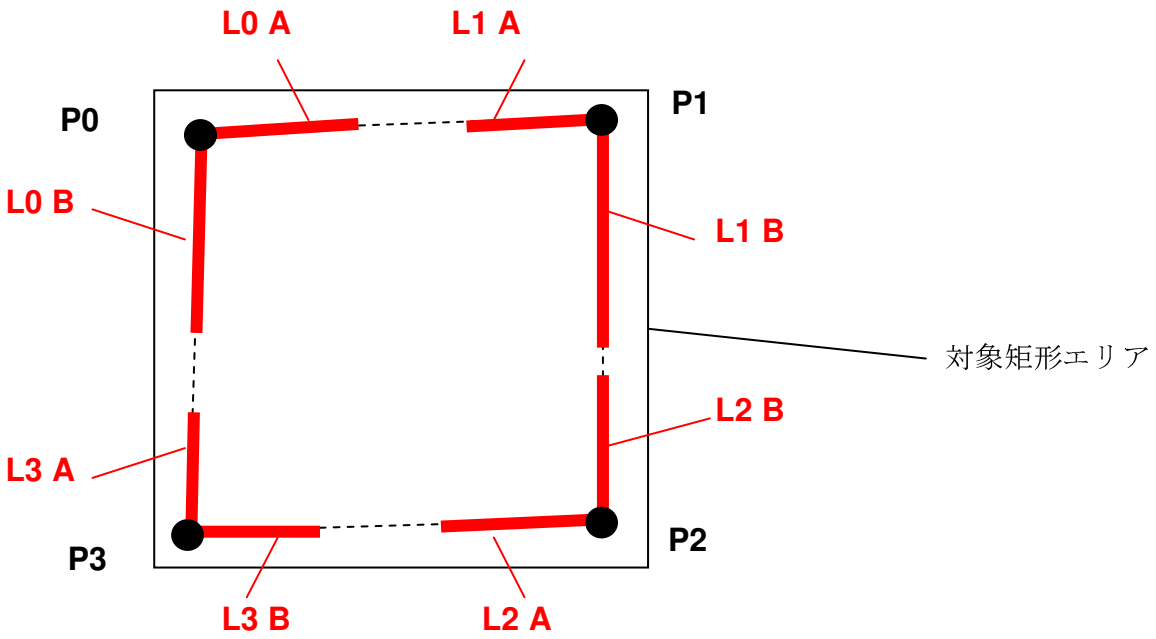
左下コーナーから対角方向に、下図のように斜めに探索し、左下コーナーポイント（P3）を定義する。



【原点の決定】

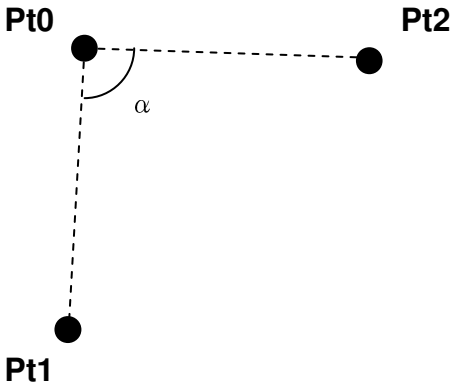
各コーナーポイントで、L を定義する。

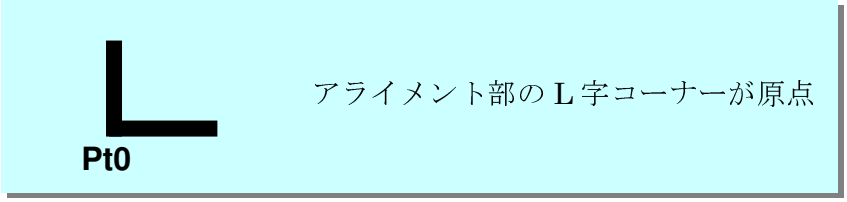
Bresenham の線分発生アルゴリズムを用いて、コーナーポイント P0~P3 を結ぶ線分上の点（座標）を生成し、その点上で、コーナーポイントからビットが連続して存在する長さを L とする。



ＬＡ＋ＬＢが最大であるコーナーポイントを原点（Pt0）とし、原点の反時計方向の点を Pt1、時計方向の点を Pt2 とする。

ここで、∠Pt1Pt0Pt2 の成す角度αは、85°＜α＜95°でなければならない。



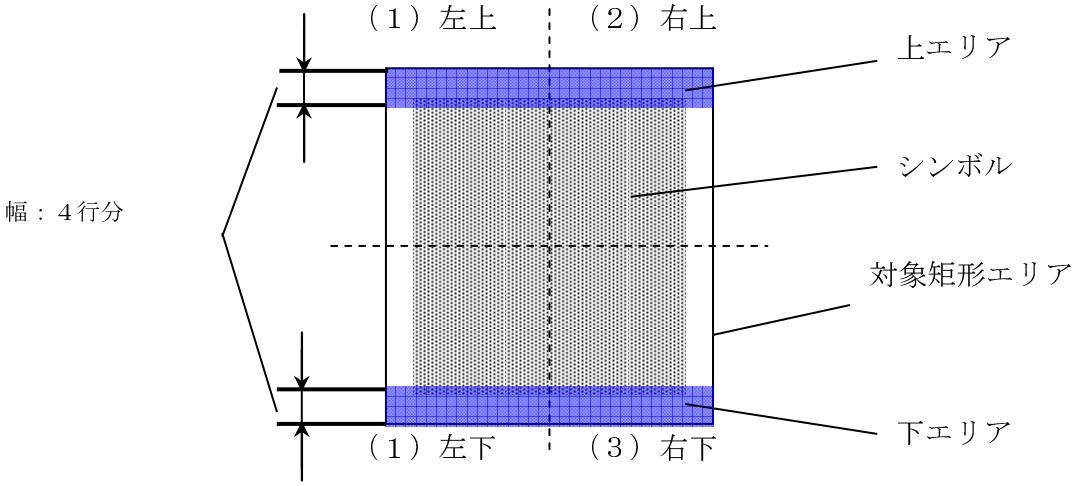


原点の検索２

【探索エリアとコーナーポイント探索】

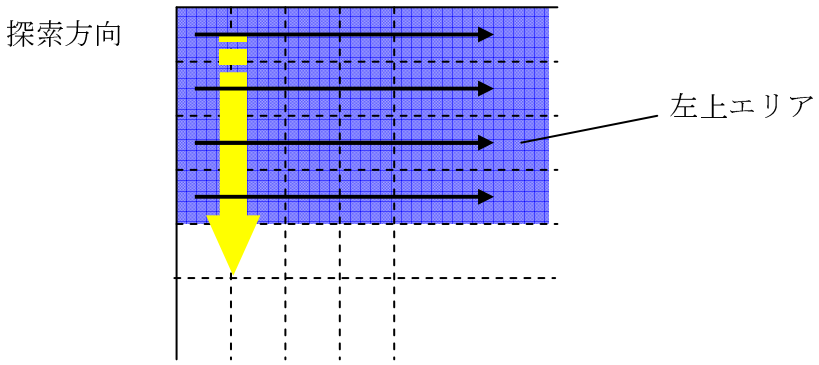
対象矩形エリアの、上下エリア（下図参照）から探索する。

（上下エリアで見つからなければ、左右エリアから探索する）



（1） 左上エリア

左上コーナーから、下図のように横方向に探索する。

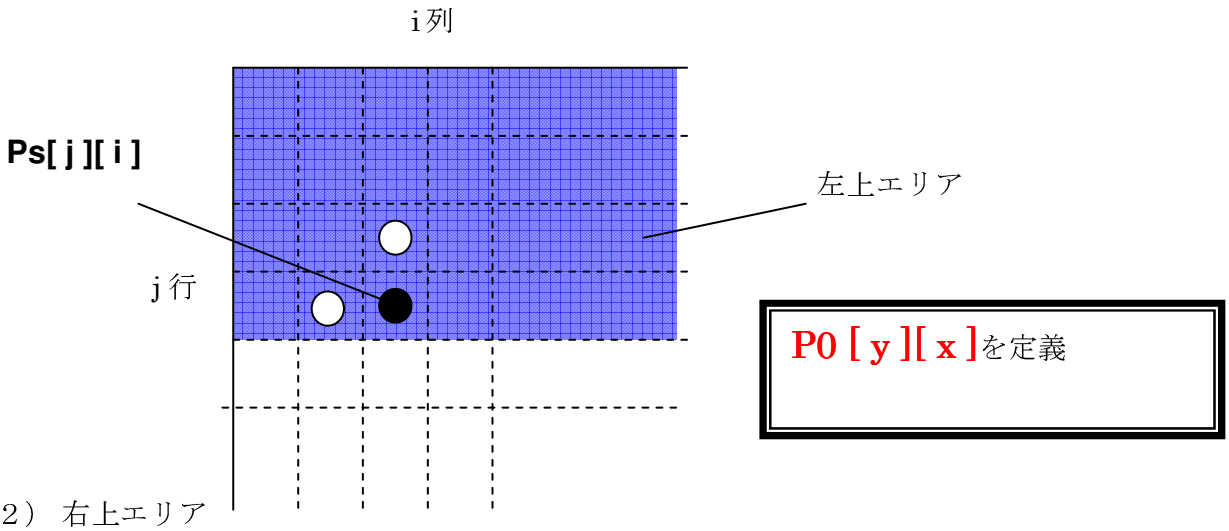


i＝0 から探索し、ビットがあれば次の行を探索する。探索エリア内で全行探索し、最後にビットが存在した点を Ps [j][i]とする。

左上コーナーポイント（P0）の横方向位置、縦方向位置を x＝i、y＝j とする。

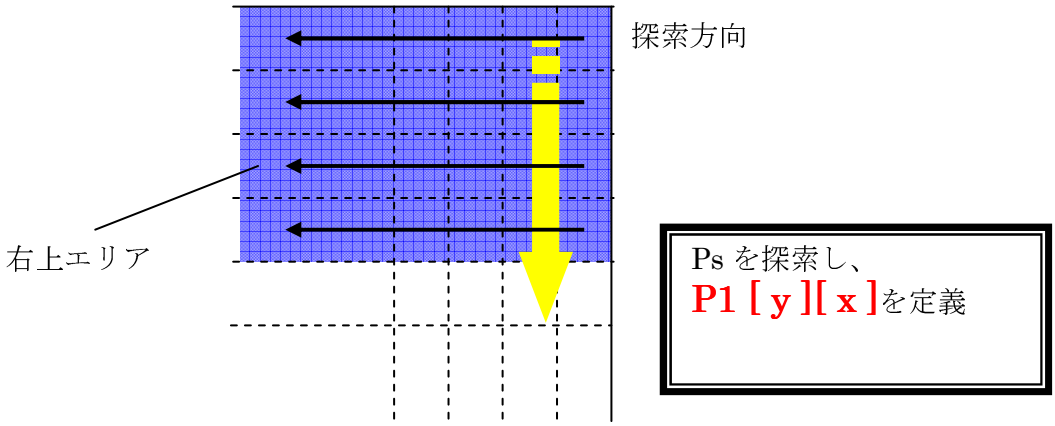
Ps [j-1][i]にビットがなければ、y＝j+1

Ps [j][i-1]にビットがなければ、x＝i+1



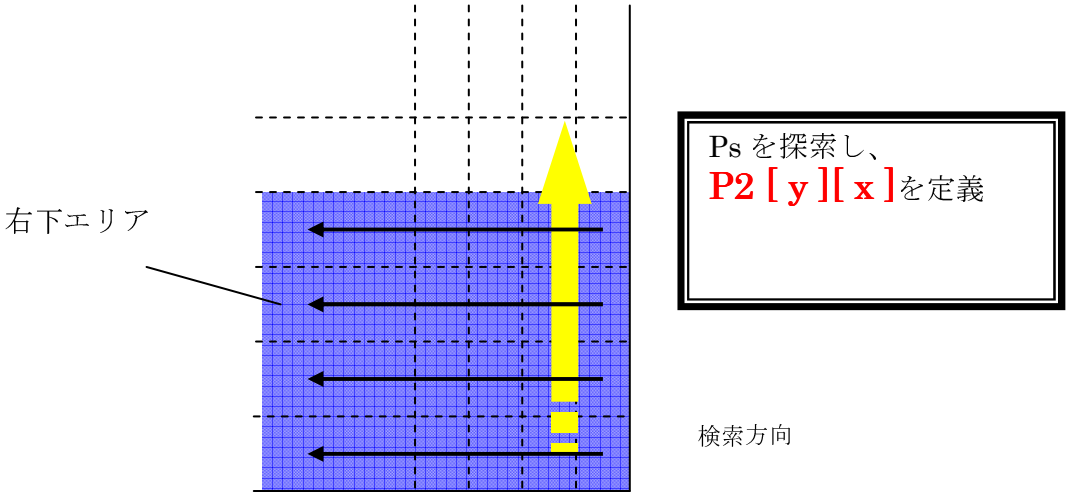
（2） 右上エリア

右上コーナーから、下図のように横方向に探索し、右上コーナーポイント（P1）を定義する。



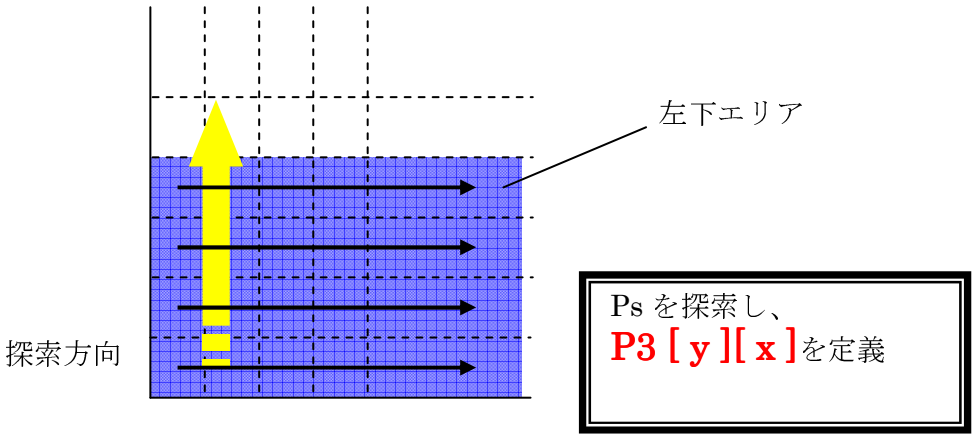
（３） 右下エリア

右下コーナーから、下図のように横方向に探索し、右下コーナーポイント（P2）を定義する。



（４） 左下エリア

左下コーナーから、下図のように横方向に探索し、左下コーナーポイント（P3）を定義する。



【原点の決定】

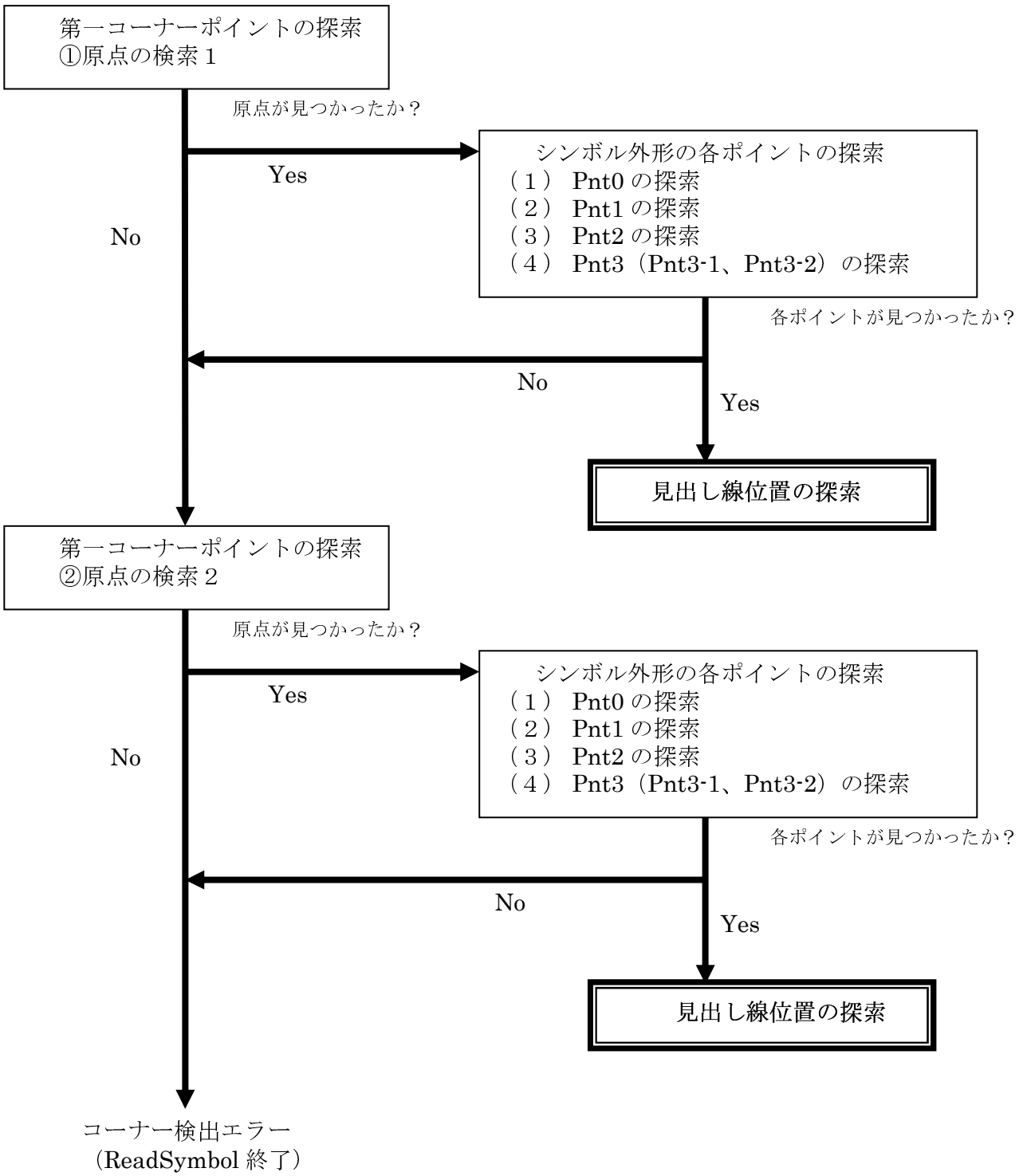
4　点の検索１【原点の決定】と同様の手順で、原点を決定する。

4-3　シンボル外形の各ポイント探索

「第一コーナーポイントの探索」で探索した点 Pt0、Pt1、Pt2 を基に、四隅の点（Pnt0、Pnt1、Pnt2、Pnt3）を探索する。

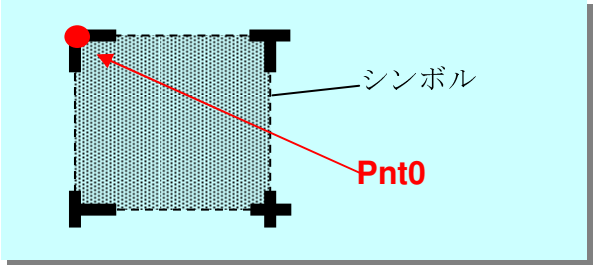
【四隅の点の探索】

シンボル外形の各ポイントは、以下の手順で探索する。



（１） Pnt0

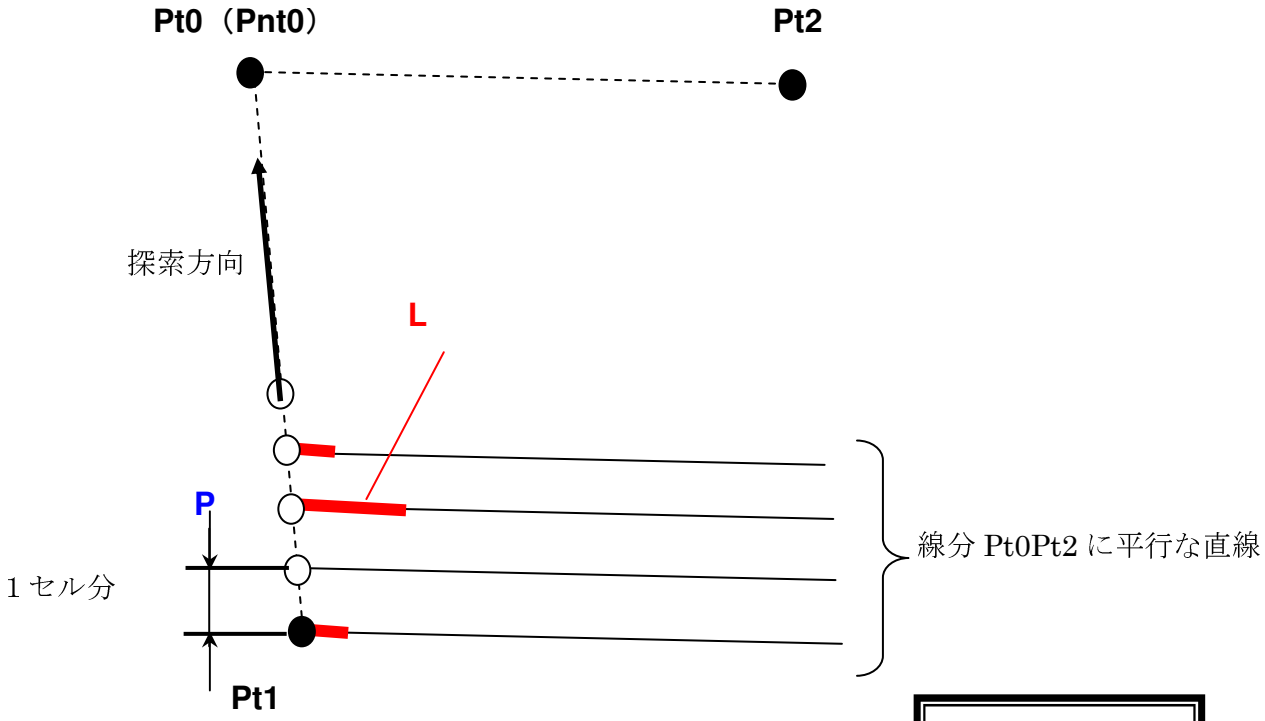
Pnt0 は、原点 Pt0 と同じ点とする。

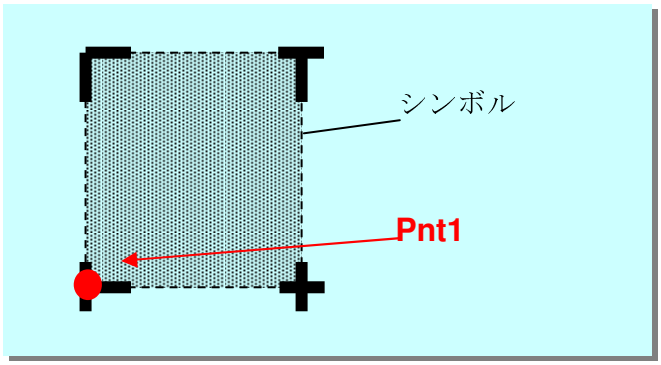


（２） Pnt1

線分 Pt1Pt0 上を Pt1 から Pt0 方向に、１セル間隔で 5 セル分探索する。

探索点 P から、線分 Pt0Pt2 に平行な直線を引き、その直線上でビットが連続して存在する長さを L とする。L が 6 セル以上であれば、探索点 P を Pnt1 とする。

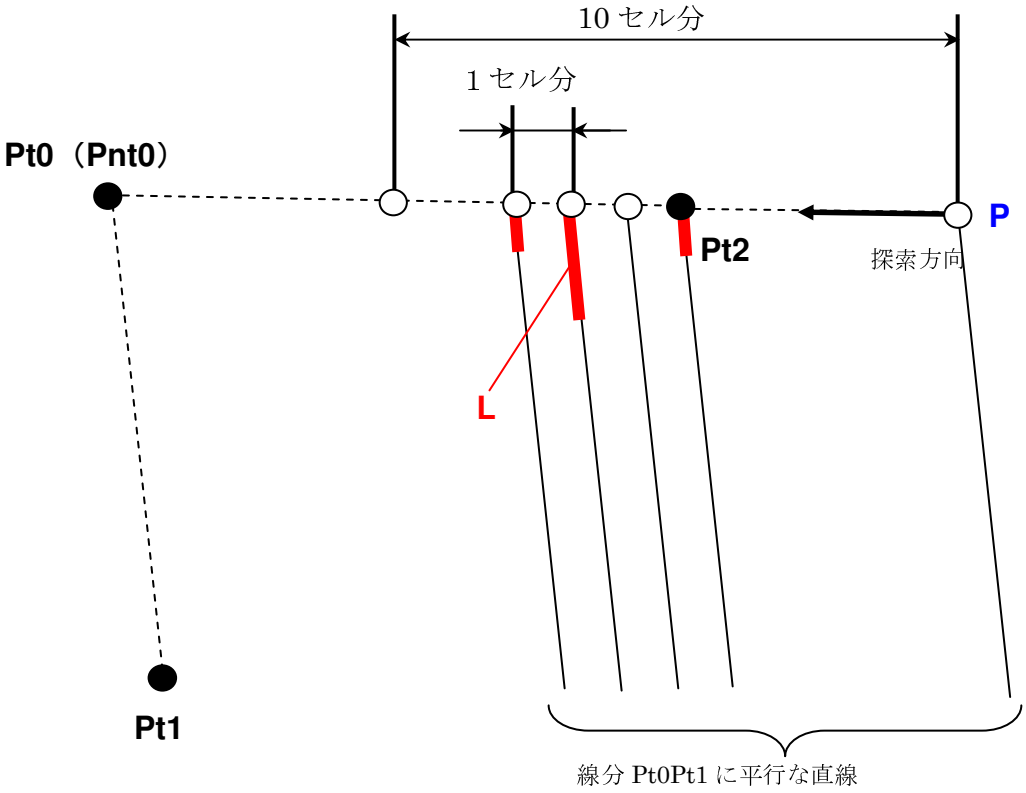


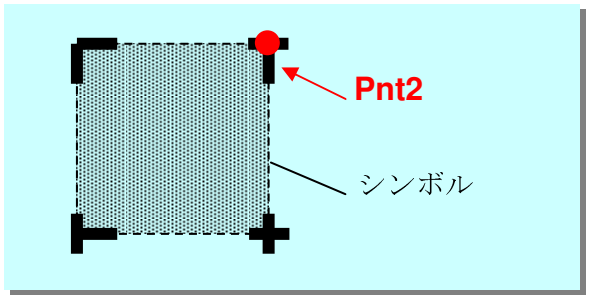


（３） Pnt2

下図のように点 P から、点 Pt2 の Pt0Pt2 方向±5 セル区間で探索する。

探索点 P から、線分 Pt0Pt1 に平行な直線を引き、その直線状でビットが連続して存在する長さを L とする。L が 6 セル以上であれば、探索点 P を Pnt2 とする。



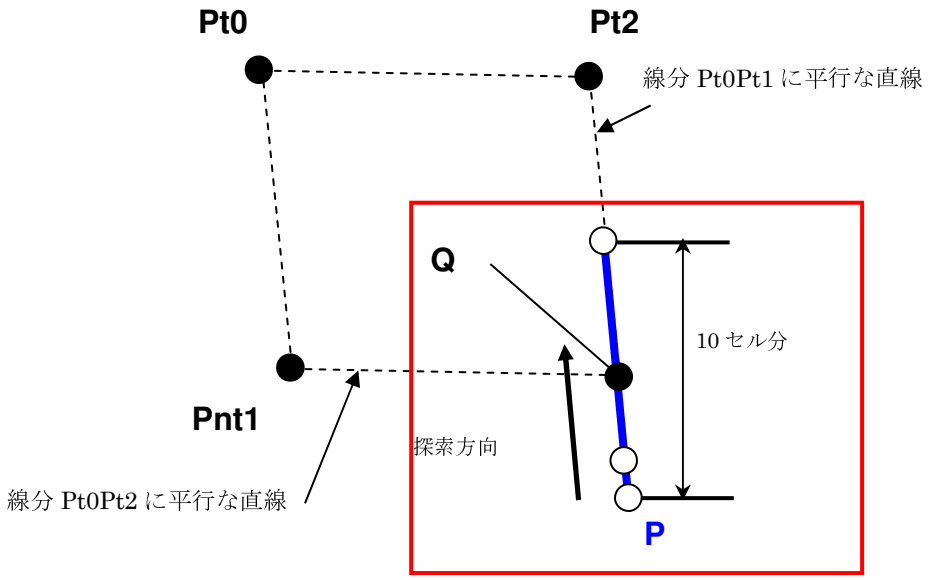


Pnt2 を定義

（４） Pnt3

Pnt1 から線分 Pt0Pt2 に平行な線分の端点 Pnt3-1 を探索する。

下図のように点 P から、点 Q の Pt2Q 方向±5 セル区間で探索する。ここで、点 Q は、Pt2 を始点とした Pt0Pnt1 と平行な直線と、Pnt1 を始点とした Pt0Pt2 に平行な直線の交点とする。

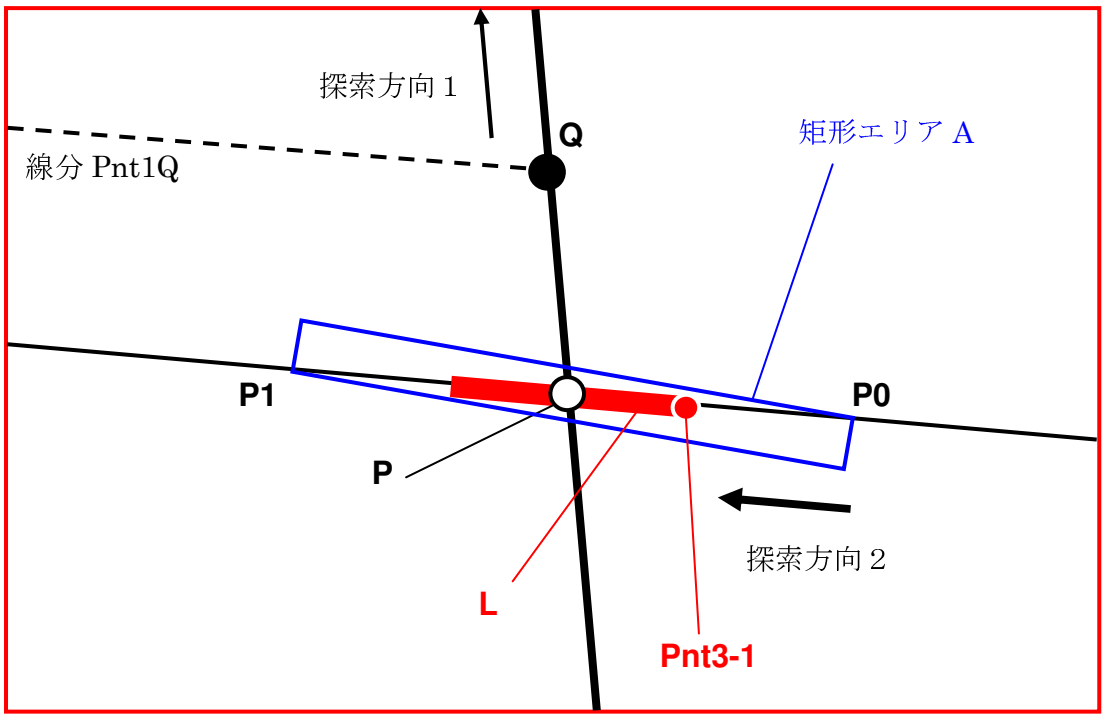


【端点の探索】

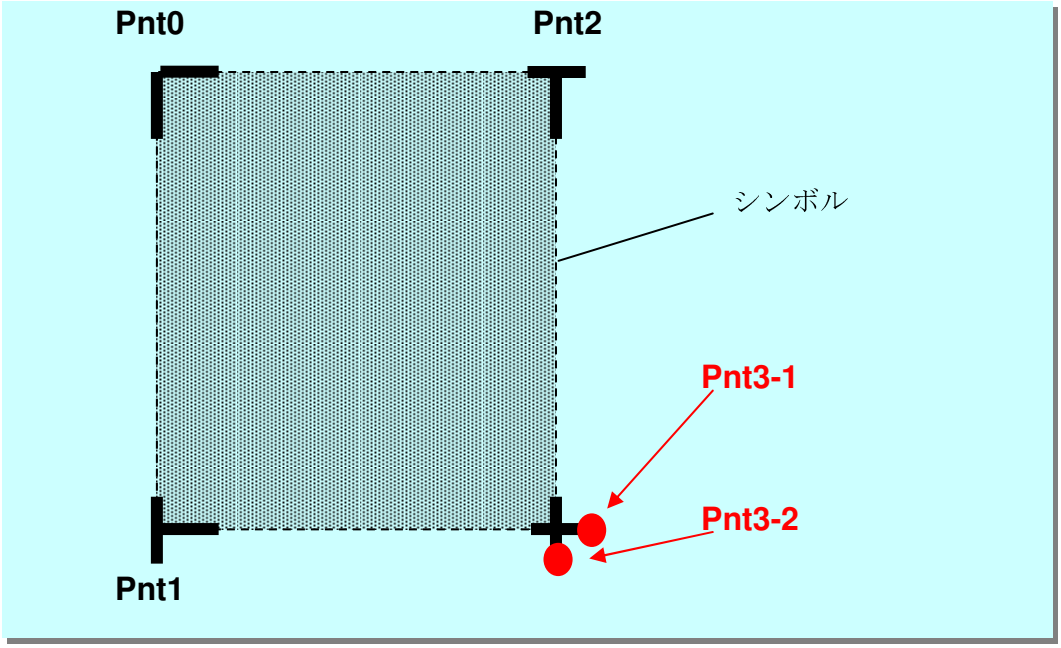
点 P から探索する（探索方向１）とき、探索点 P を中心に矩形エリア A を設定し、エリア内で端点 Pnt3-1 を探索する。

ここで、矩形エリア A は、点 P を通り線分 Pnt1Q と平行な直線上で、点 P を中心に長さが 8.5 セルとなる点 P0、P1 を対角点とするエリアである。

矩形エリア A の頂点 P0 から線分 P0P1 上を探索し（探索方向２）、その線分上でビットが連続して存在する最大長区間の長さを L とする。L が 4 セル以上であれば、連続開始点を Pnt3-1 とする。



Pnt2 から線分 Pt0Pt1 に平行な線分の端点 Pnt3-2 を探索する。
（探索手順は、Pnt3-1 の探索時と同様）



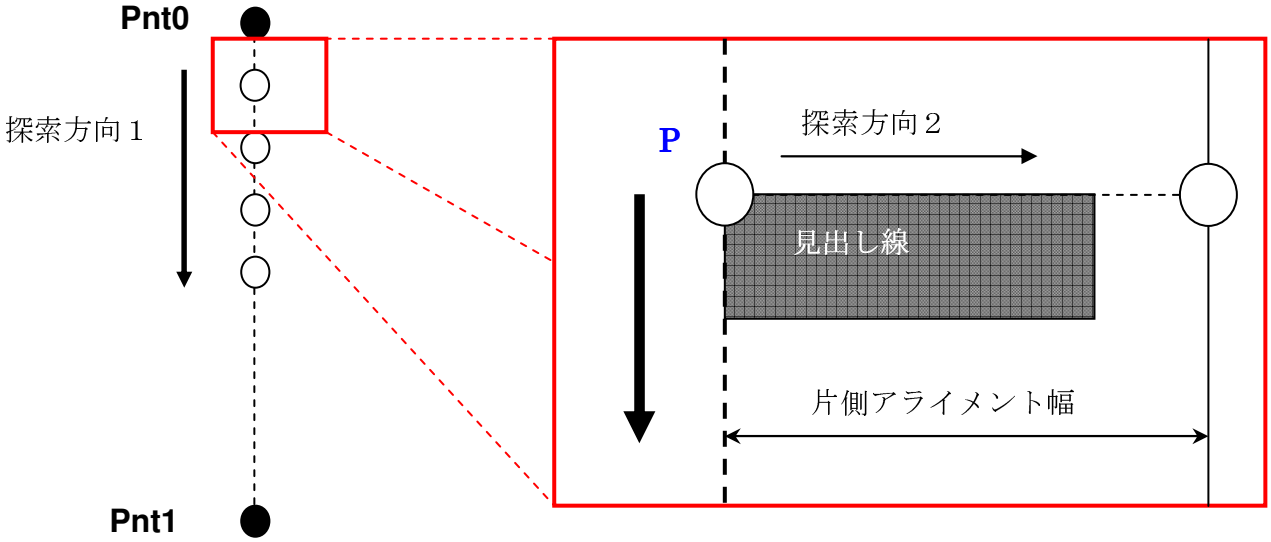
4-4　見出し線位置の探索

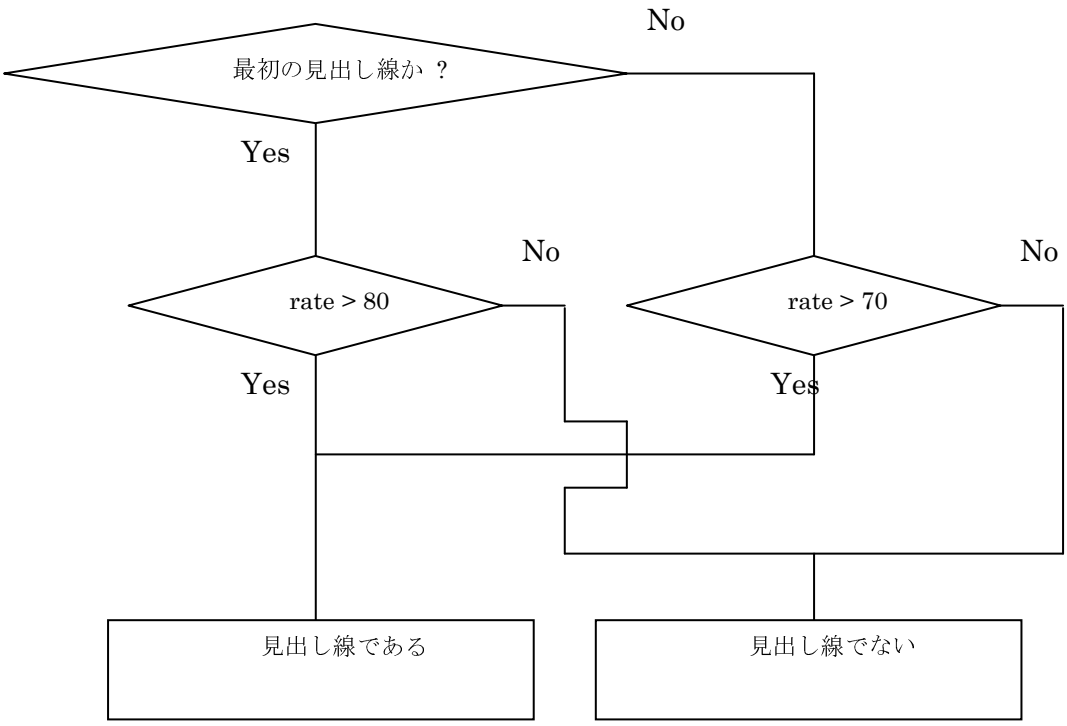
「シンボル外形の各ポイント探索」で探索した点 Pnt0、Pnt1、Pnt2、Pnt1-3、Pnt2-3 を基に、ユニットの見出し線位置を探索する。

① 列見出し線位置の探索

線分 Pnt0Pnt1 上を Pnt0 から Pnt1 方向に探索し（探索方向１）、左手垂直方向に見出し線を探索する（探索方向２）。同様に、線分 Pnt2Pnt2-3 上を Pnt2 から Pnt2-3 方向に探索し、左手垂直方向に見出し線を探索する。

Bresenham の線分発生アルゴリズムを用いて、P、Q を結ぶ線分 PQ 上の点（座標）を生成し、その点上のビットの割合 rate(%)を計算する。ビットの割合により見出し線を判定する。

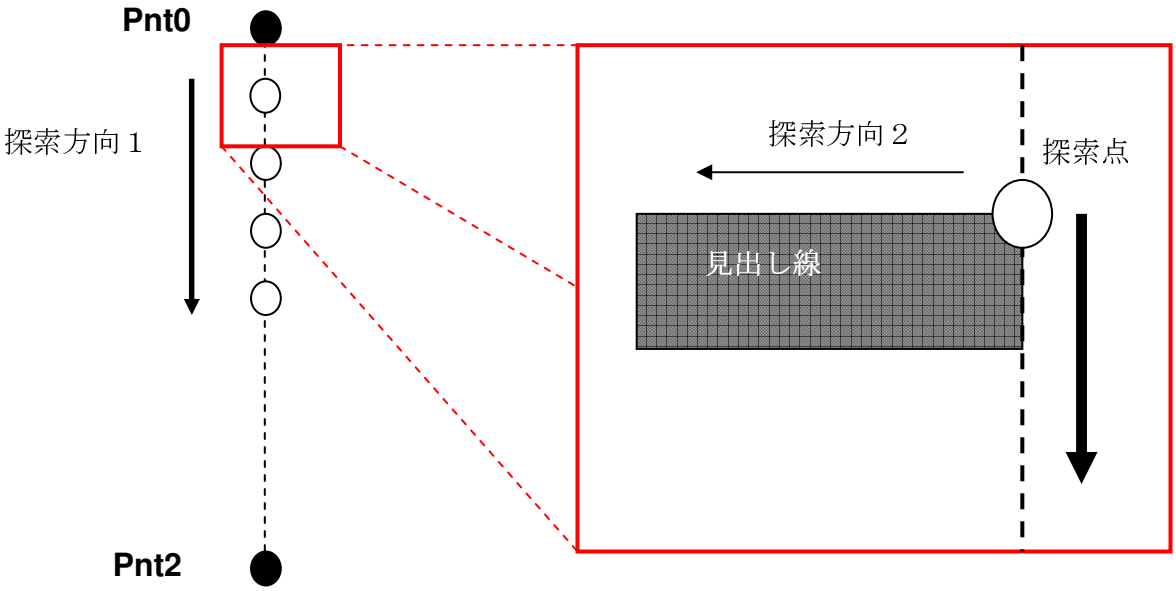


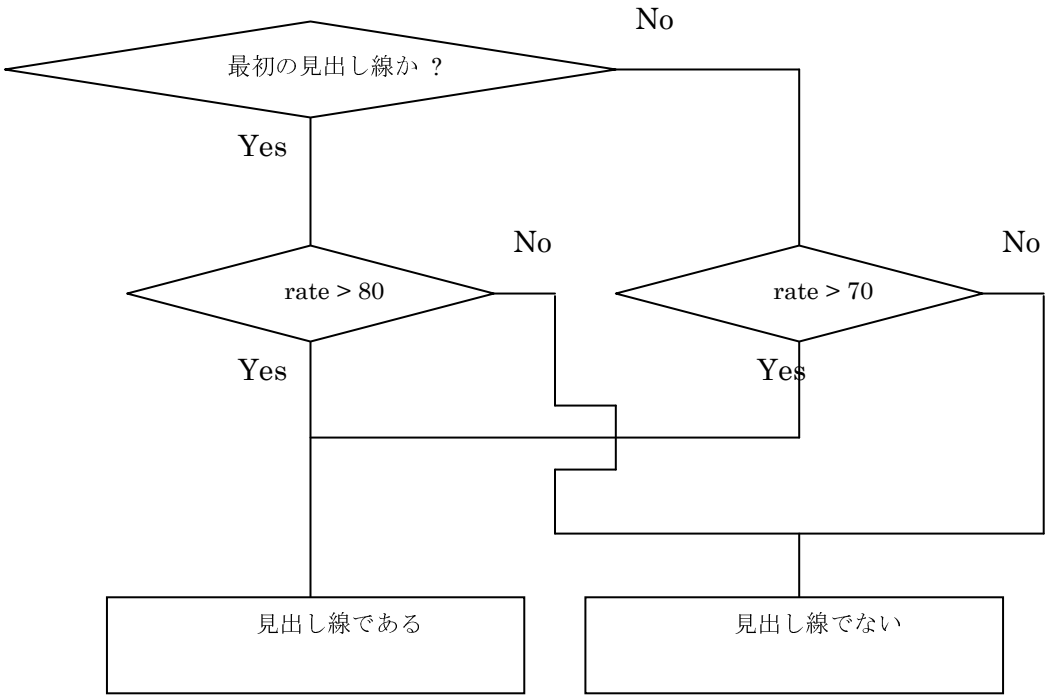


② 行見出し線位置の探索

線分 Pnt0Pnt2 上を Pnt0 から Pnt2 方向に探索し（探索方向１）、右手垂直方向に見出し線を探索する探索方向２）。同様に、線分 Pnt1Pnt1-3 上を Pnt1 から Pnt1-3 方向に探索し、右手垂直方向に見出し線を探索する。

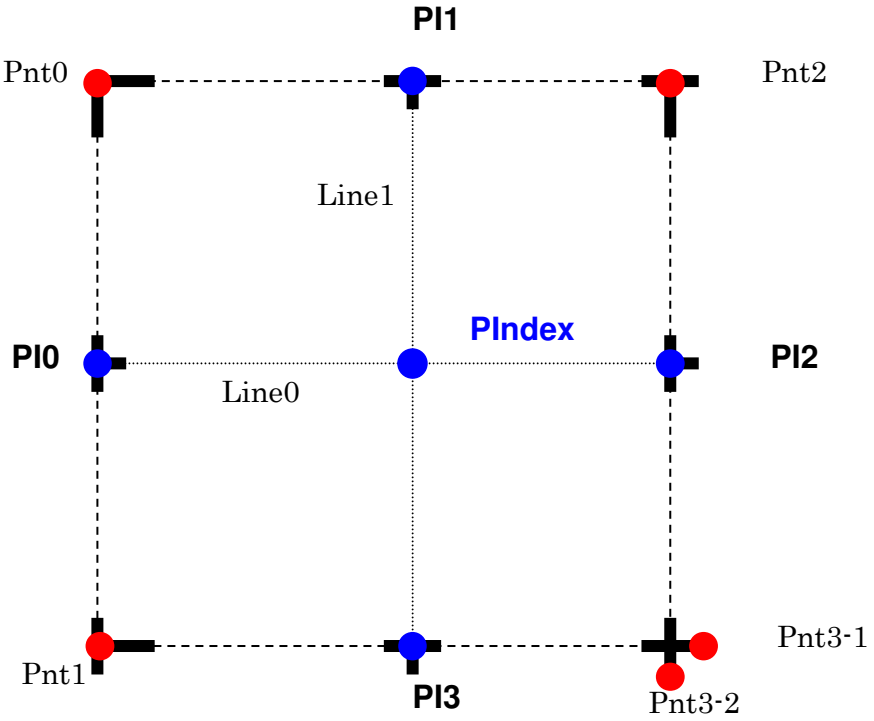
見出し線の判定方法は、①列見出し線位置の探索と同様である。





4-5　交差点の計算

「見出し線位置の探索」で探索した見出し線位置（PI1、PI2、PI3、 PI4）から、各見出し線の交点 PIndex を計算する。



【Pindex の計算】

見出し位置 Pnt0、Pnt1、Pnt2、Pnt3 から交点 Pindex(X,Y)を求める。

2 点 Pnt0(x0,y0)、Pnt2(x2,y2)を通る直線 Line0 及び 2 点 Pnt1(x1,y1)、Pnt3(x3,y3)を通る直線 Line1 を求める。

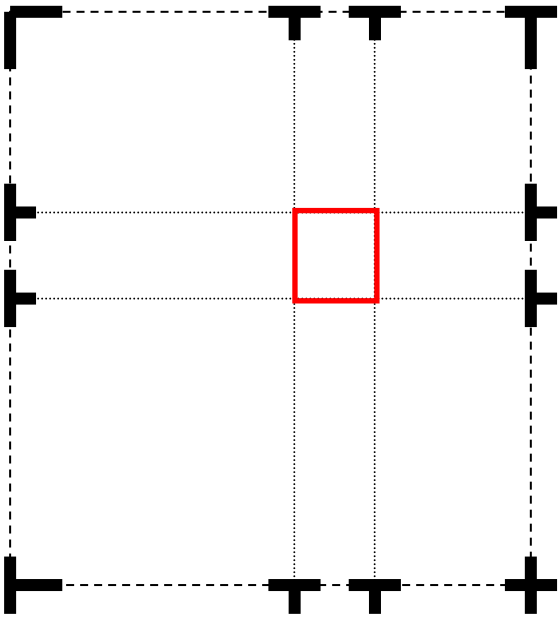
$$\begin{cases} y = a0\ x + b0 & \text{(Line0)} \\ y = a1\ x + b1 & \text{(Line1)} \end{cases}$$

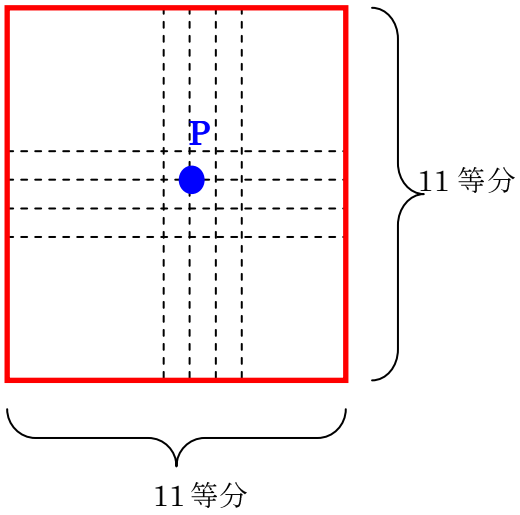
Line0 と Line1 の交点 Pindex(X,Y)を求める。

$$\begin{cases} X = (b1 - b0) / (a1 - a0) \\ Y = a0\ X + b0 \end{cases}$$

4-6　ビットサンプリング

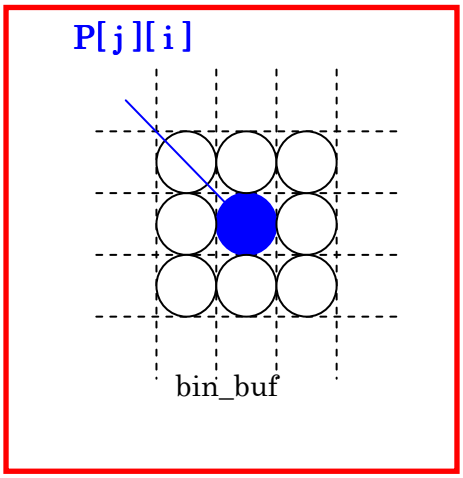
各見出し線により縦横に囲われたデータブロック（1 ユニット）内を、等分に分割して各セルの中心座標 P を求める。



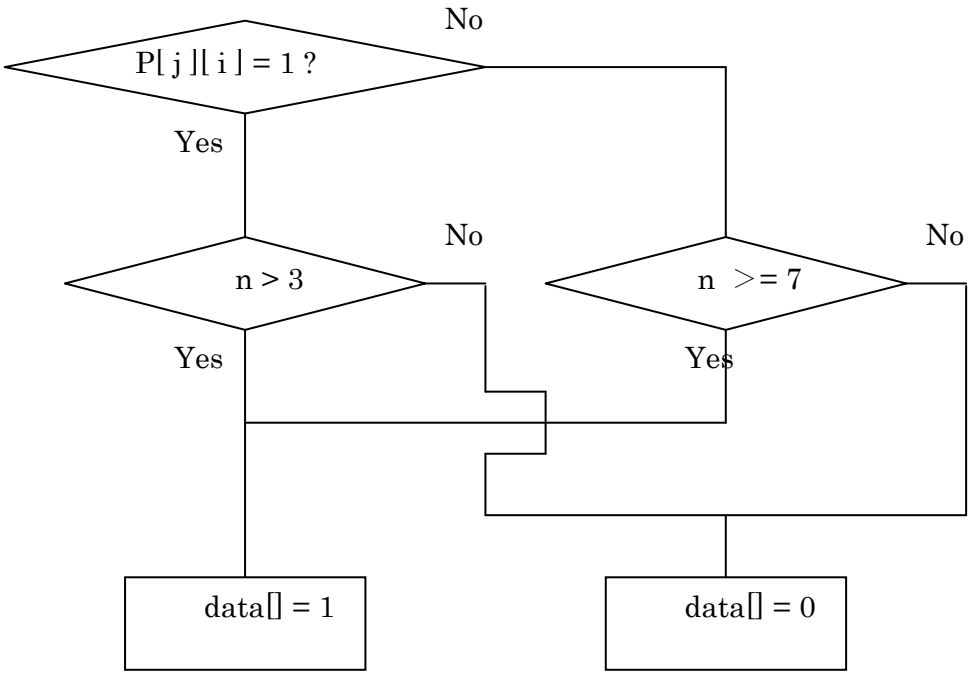


1 ユニット（11×11）

3 のピクセルを読み出し、各ピクセルに重みを付け 1 or 0 の判定を行い、ビット列データ data[] を作成する。



1 ユニット（11×11）

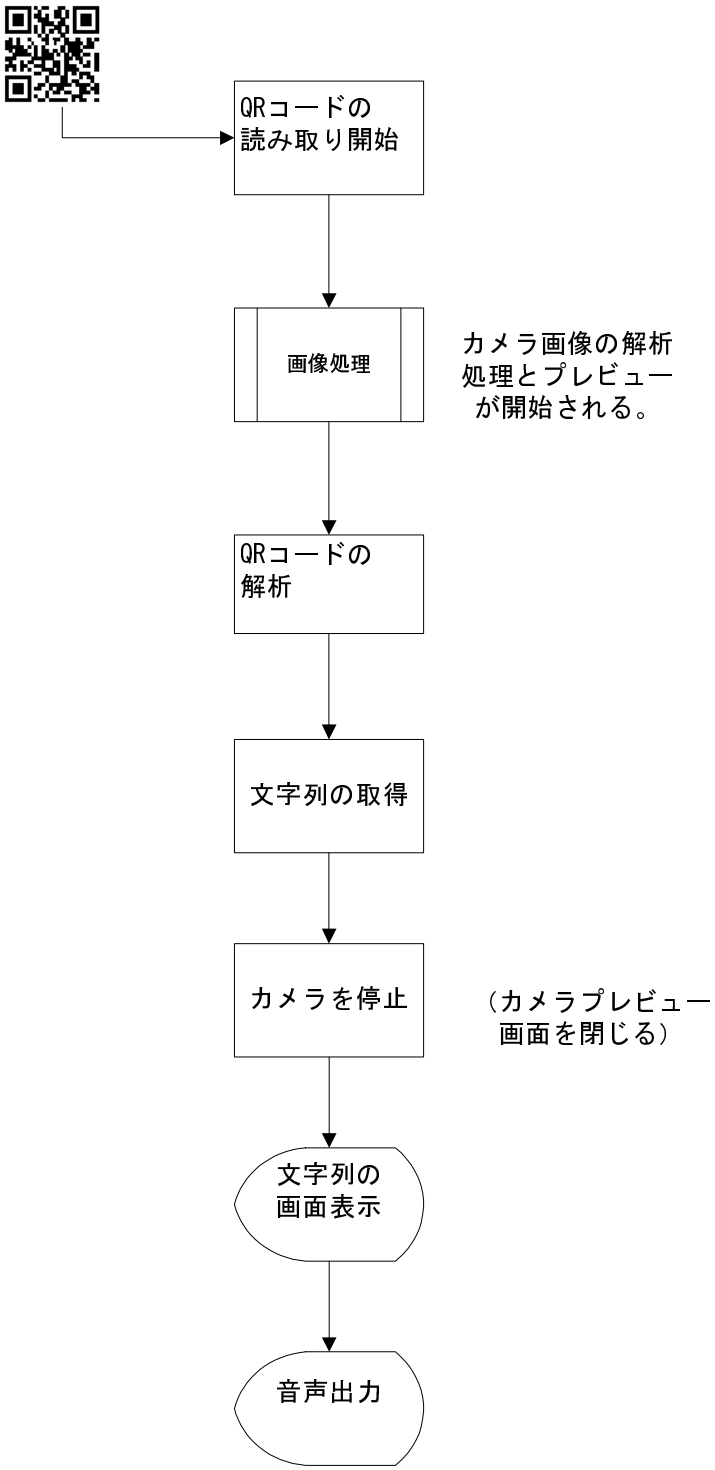


n ：3×3 ピクセル内のビット数

5. QR コードの取り込み

ここでは、新たに追加した QR コードを取り込み、解析し文字列を画面に表示、音声出力までの流れを説明する。

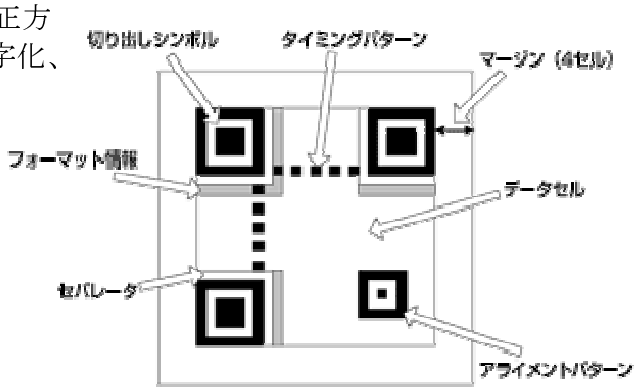
5-1　処理の流れ



5-2　QR コードの読み取り

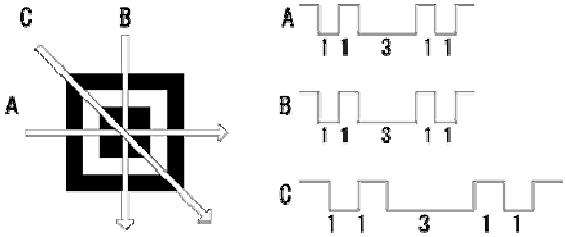
QR コードの切り出しシンボルを検出しフォーマット情報を読みデコードする。

QR コードを構成する最小の単位であるセル(白黒の正方形)の組み合わせで現わされているものを読み取り文字化、音声化させる。



QR コードの構成

○切り出しシンボル



3 コーナー（A,B,C）に配置される部分を検索し、位置を正確に認識させる。

○タイミングパターン
シンボル内のモジュール座標を認識し解析する。

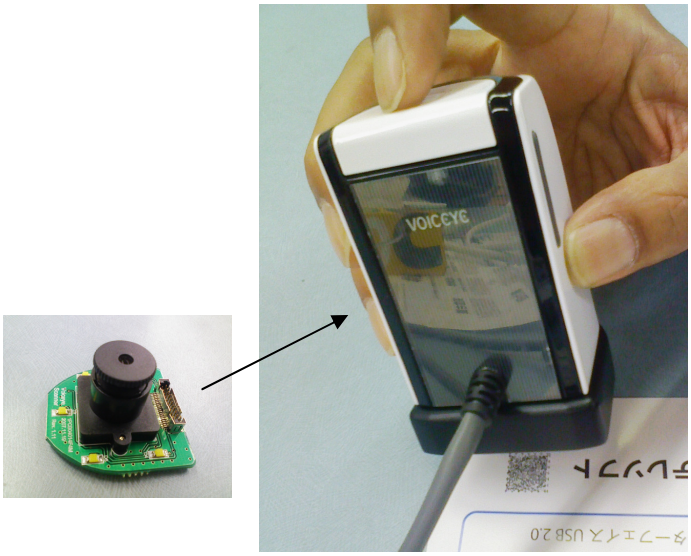
○フォーマット情報
まず最初にここを読み誤り訂正率、マスクパターンを認識する。

5-3　QR コードの解析

QR コードの切り出しシンボルを検出しフォーマット情報、タイミングパターン、セルを抽出し解析して文字変換したものを画面に表示、音声出力を行う事で、今回の QR コード読み取り機能を追加した。

1　PC 接続型

仕様
カメラ：CMOS
I/F　　：USB　2.0(コード 1.5m)
サイズ：85mm（H）
　　50mm（W）
　　52mm（D）



PC アプリケーション

初期メニュー

ここでは、まず音声コード読み上げプログラムをアイコンから起動、或いは、プログラムから、または、スタートアップから Windows 起動時に立ち上がるように設定し、２種類のプログラムを選択するメニュー画面を表示させる。