(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2010-44586
(P2010-44586A)
(43)公開日　平成22年2月25日(2010.2.25)

(51)Int.Cl.		F I			テーマコード（参考）
G06T 3/00	(2006.01)	G06T	3/00	200	5B035
G06K 7/10	(2006.01)	G06K	7/10	P	5B057
G06K 19/06	(2006.01)	G06K	19/00	E	5B072
H04N 1/40	(2006.01)	H04N	1/40	101Z	5C076
H04N 1/387	(2006.01)	H04N	1/387		5C077

審査請求 未請求　請求項の数 7　ＯＬ　（全 24 頁）

(21)出願番号　特願2008-208095 (P2008-208095)
(22)出願日　平成20年8月12日 (2008.8.12)

（特許庁注：以下のものは登録商標）
１．ＱＲコード

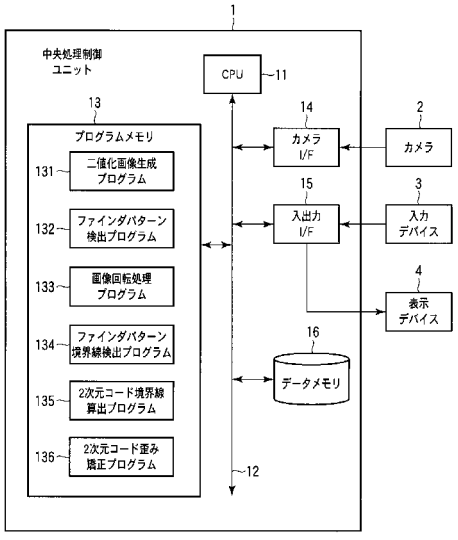
(71)出願人　000004226
日本電信電話株式会社
東京都千代田区大手町二丁目３番１号
(74)代理人　100058479
弁理士　鈴江　武彦
(74)代理人　100108855
弁理士　蔵田　昌俊
(74)代理人　100091351
弁理士　河野　哲
(74)代理人　100080285
弁理士　小出　俊實
(74)代理人　100087963
弁理士　石川　義雄
(74)代理人　100075672
弁理士　峰　隆司

最終頁に続く

(54)【発明の名称】２次元コード読取装置とそのプログラム

(57)【要約】
【課題】　２次元コード画像が曲線状歪みを有する場合でも、この曲線状歪みを効果的に矯正して高精度のコード解読を可能にする。
【解決手段】　カメラ２により撮像された２次元コードの画像データを二値化画像に変換した後、当該二値化画像から２次元コードのファインダパターンを検出し、この検出されたファインダパターンの上下左右各辺の座標値を検出する。そして、この検出されたファインダパターンの上下左右各辺の座標値をもとに、上記２次元コードの画像データの上下左右各辺の形状を直線及び曲線で近似した近似線を生成し、この生成された近似線をもとに上記２次元コードの画像データの歪みを矯正して、この歪みが矯正された２次元コードの画像データをもとに上記２次元コードを解読する。
【選択図】　図１

図１



【特許請求の範囲】
【請求項１】
撮像手段から２次元コードの画像データを取り込んでメモリに記憶する手段と、
前記メモリに記憶された２次元コードの画像データを読み出し、この読み出した画像データから当該２次元コードに含まれる複数のファインダパターンを検出するファインダパターン検出手段と、
前記検出された複数のファインダパターンの各々についてその輪郭を表す辺の境界点を検出するファインダパターン境界検出手段と、
前記検出された複数のファインダパターンの辺の境界点をもとに、前記２次元コードの画像データの輪郭を表す４辺の形状を直線及び曲線の少なくとも一方で近似した近似線を生成する２次元コード境界線検出手段と、
前記生成された近似線をもとに前記２次元コードの画像データの歪みを矯正し、歪みが矯正された２次元コードの画像データを前記メモリに記憶する２次元コード歪み矯正手段と、
前記歪みが矯正された２次元コードの画像データをもとに前記２次元コードを解読し、その解読データを出力する解読手段と
を具備することを特徴とする２次元コード読取装置。
【請求項２】
前記２次元コード境界線検出手段は、
前記ファインダパターン境界検出手段により検出された複数のファインダパターンの辺の境界点をもとに、前記２次元コードの画像データの輪郭を表す辺の傾きを算出する手段と、
前記算出された辺の傾きから推測される方向に画素探索を行って、前記２次元コードの辺を表す境界点を検出する手段と、
前記検出された境界点に含まれるエラー点を除去する手段と
を備えることを特徴とする請求項１記載の２次元コード読取装置。
【請求項３】
前記辺の傾きを算出する手段は、第１及び第２のファインダパターンの各辺を表す境界点をもとにそれぞれ当該各辺の傾きを算出し、この算出された第１及び第２のファインダパターンの各辺の傾きをもとに、前記２次元コードの画像データの一辺の傾きを推測することを特徴とする請求項２記載の２次元コード読取装置。
【請求項４】
前記エラー点を除去する手段は、前記検出された境界点の各々について、当該境界点が、前記算出された辺の傾きから推測される範囲内に含まれているか否かを判定し、含まれていないと判定された境界点をエラー点として前記境界点から削除することを特徴とする請求項２記載の２次元コード読取装置。
【請求項５】
前記エラー点を除去する手段は、前記２次元コードの画像データの４辺のうち直交する第１及び第２の辺の交点を算出する手段と、この算出された交点より外側の範囲に位置する境界点をエラー点として削除する手段とを備え、
前記２次元コードの画像データの第１及び第２の辺の交点を算出する手段は、
第１の辺を表す境界点の集合に含まれる１個以上の境界点を算出基準点としてしきい値線を引き、第２の辺を表す境界点集合に含まれる境界点が前記しきい値線より交点側に位置するか否かを判定する手段と、
前記第２の辺を表す境界点の集合のうち前記しきい値線より交点側に位置しない境界点の中から、当該交点に最も近い境界点をしきい値外近接点として検出する手段と、
前記算出基準点から前記第１の辺の傾きを推定して引いた直線と、前記第２の辺のしきい値外近接点から第２の辺の傾きを推定して引いた直線との交点をもとに、交点候補点を算出する手段と、
前記第１の辺を表す境界点の集合に含まれる境界点のうち前記しきい値線より交点側

に位置する境界点の中から前記しきい値直線に最も近い境界点をしきい値内近接点として検出すると共に、前記第２の辺を表す境界点集合に含まれる境界点のうち前記しきい値線より交点側に位置する境界点の中から前記しきい値直線に最も近い境界点をしきい値内近接点として検出する手段と、

前記検出された第１の辺のしきい値内近接点と、前記検出された第２の辺のしきい値内近接点のうち、前記算出された交点候補点より外側にあるものはどれかを判定する手段と

を有することを特徴とする請求項２記載の２次元コード読取装置。

【請求項６】

前記２次元コード歪み矯正手段は、

２次元コードの画像データの各頂点を結ぶ頂点フレームを生成する頂点フレーム生成手段と、

前記矯正対象の２次元コードの画像データにおける目標座標のピクセル値を算出する際に、前記生成された頂点フレームの、上辺における前記目標座標に対応する点の座標値と、下辺における前記目標座標に対応する点の座標値と、左辺における前記目標座標に対応する点の座標値と、右辺における前記目標座標に対応する点の座標値をそれぞれ算出する頂点フレーム対応点算出手段と、

前記２次元コードの画像データの、上辺における前記目標座標に対応する境界点の座標値と、下辺における前記目標座標に対応する境界点の座標値と、左辺における前記目標座標に対応する境界点の座標値と、右辺における前記目標座標に対応する境界点の座標値をそれぞれ算出する境界対応点算出手段と、

前記頂点フレームにおいて前記目標座標に対応する目標座標値を算出する頂点フレーム内目標座標値算出手段と、

前記頂点フレームの上下左右の各辺について算出された前記目標座標に対応する点から、前記２次元コードの画像データの上下左右の各辺について算出された前記目標座標に対応する境界点へのベクトルを重み付け加算して、２次元コード歪みベクトルを算出する２次元コード歪み算出手段と、

前記頂点フレームについて算出された目標座標値に、前記算出された２次元コード歪みベクトルを加算して、前記２次元コードの画像データにおける前記目標座標に対応する目標座標値を算出する２次元コード内目標座標値算出手段と

を

備えることを特徴とする請求項１記載の２次元コード読取装置。

【請求項７】

前記請求項１乃至６のいずれか記載の２次元コード読取装置において、各手段の処理を実行するために使用されるプログラム。

【発明の詳細な説明】

【技術分野】

【０００１】

この発明は、例えば印刷された２次元コードをカメラにより撮像し、その画像データをもとに２次元コードを解読する２次元コード読取装置とそのプログラムに関する。

【背景技術】

【０００２】

近年、雑誌等の印刷物に表示されたＱＲコードに代表される２次元コードを例えば携帯電話機のカメラにより撮像し、その画像データから上記２次元コードに含まれるＵＲＬ（Uniform Resource Locator）を認識する。そして、このＵＲＬをもとにインターネットに対しアクセスすることにより、対応するＷｅｂサーバから情報を取得できるようにするサービスが普及している。この種のサービスを利用すると、ユーザはＵＲＬをキー操作により入力する必要がなくなり、大変便利である。

【０００３】

しかしながら、２次元コードを携帯電話機のカメラにより撮像する際に、２次元コード

にカメラが正対していないと、２次元コードの撮像画像が台形状に歪んでしまい、２次元コードを正しく認識できなくなることがある。そこで、従来では撮像された２次元コードの形状を矯正する機能を備えた２次元コード読取装置が提案されている。例えば、図１９に示すように撮像された２次元コード画像ＰＣの周囲に四角形ＡＢＣＤを設定し、０≦ｓ≦１，０≦ｔ≦１を満たすｓ、ｔに対してＡＰ：ＰＢ＝ＣＰ′：Ｐ′Ｄ＝ｓ：１－ｓ、ＡＱ：ＱＣ＝ＢＱ′：Ｑ′Ｄ＝ｔ：１－ｔとなるＰＰ′及びＱＱ′を設定する。これにより任意のｓ及びｔに対してＰＰ′とＱＱ′の交点Ｒ(s,t) が求まる。ここで、あるｔに対して０から１までｓを変化させることにより２次元コードの一部を検査することが可能となる。さらに、０から１までのｔに対して上記検査線を設定することにより２次元コード全体を検査することが可能となり、画像に含まれるデータ領域のセル位置が決定される（例えば、特許文献１を参照。）。

【０００４】

【特許文献１】特開２００６－１５５２１８号公報

【発明の開示】

【発明が解決しようとする課題】

【０００５】

ところが、上記従来より提案されている装置は、台形状の歪みに対しては有効であるが、例えばカメラの光学系の歪みや印刷物の表面の歪みの影響により、２次元コードの撮像画像が例えば図１８（ａ），（ｂ）に示すように糸巻き型又は樽型に歪んだ場合には、このような曲線状歪みを補正することができず、その結果高精度の認識を行うことができなかった。

【０００６】

この発明は上記事情に着目してなされたもので、その目的とするところは、２次元コード画像が曲線状歪みを有する場合でも、この曲線状歪みを効果的に矯正して高精度のコード解読を可能にした２次元コード読取装置とそのプログラムを提供することにある。

【課題を解決するための手段】

【０００７】

上記目的を達成するためにこの発明の一観点は、撮像手段から２次元コードの画像データを取り込んでメモリに記憶する。そして、メモリから読み出した画像データから、先ず当該２次元コードに含まれる複数のファインダパターンを検出し、この検出された複数のファインダパターンの各々についてその輪郭を表す辺の境界点を検出する。次に、この検出された複数のファインダパターンの辺の境界点をもとに、上記２次元コードの画像データの輪郭を表す４辺の形状を直線及び曲線の少なくとも一方で近似した近似線を生成し、この生成された近似線をもとに上記２次元コードの画像データの歪みを矯正して、この歪みが矯正された２次元コードの画像データを上記メモリに記憶する。そして、この歪みが矯正された２次元コードの画像データをもとに上記２次元コードを解読して出力するように構成したものである。

【０００８】

したがって、撮像した２次元コードの画像が例えば糸巻き型又は樽型の曲線状歪みを有していても、２次元コードの画像データの輪郭を表す４辺の形状が直線及び曲線の少なくとも一方により近似され、その近似線をもとに上記２次元コードの画像データの歪みが矯正される。このため、上記糸巻き型又は樽型の曲線状歪みを有する２次元コード画像に対しても高精度のコード解読を行うことが可能となる。

【０００９】

さらにこの発明は、以下のような具体的な構成を備えることを特徴とする。

第１の構成は、上記２次元コードの画像データの４辺の形状を近似して当該４辺の境界線を検出する際に、複数のファインダパターンの辺の境界点をもとに上記２次元コードの画像データの輪郭を表す辺の傾きを算出し、この算出された辺の傾きから推測される方向に画素探索を行って、上記２次元コードの画像データの辺を表す境界点を検出する。そして、上記算出された辺の傾きをもとに、上記検出された境界点に含まれるエラー点を除去

する。
より具体的には、上記２次元コード画像の辺の傾きを算出する際に、第１及び第２のファインダパターンの各辺を表す境界点をもとにそれぞれ当該各辺の傾きを算出し、この算出された第１及び第２のファインダパターンの各辺の傾きをもとに、上記２次元コードの画像データの一辺の傾きを推測する。
【００１０】
このようにすると、各ファインダパターンの境界点をもとに当該ファインダパターン間の２次元コード画像の辺の傾きが推測され、この傾きをもとに当該辺の境界点が探索される。そして、この探索された境界点の中からエラー点が除去される。このため、２次元コード画像の辺の境界線を精度良く推測できる。
【００１１】
第２の構成は、上記エラー点を除去する際に、上記検出された境界点の各々について、当該境界点が、上記算出された辺の傾きから推測される範囲内に含まれているか否かを判定し、含まれていないと判定された境界点をエラー点として上記境界点から削除するものである。
【００１２】
第３の構成は、上記エラー点を除去する際に、２次元コードの画像データの４辺のうち互いに直交する第１及び第２の辺の交点を算出し、この交点より外側の範囲に位置する境界点をエラー点として削除するものである。上記交点の算出手段としては、先ず第１の辺を表す境界点集合に含まれる１個以上の境界点を算出基準点としてしきい値線を引き、第２の辺を表す境界点集合に含まれる境界点が上記しきい値線より交点側に位置するか否かを判定する。そして、上記第２の辺を表す境界点集合のうち上記しきい値線より交点側に位置しない境界点の中から、当該交点に最も近い境界点をしきい値外近接点として検出する。次に、上記算出基準点から上記第１の辺の傾きを推定して引いた直線と上記第２の辺のしきい値外近接点から第２の辺の傾きを推定して引いた直線との交点をもとに交点候補点を算出し、上記第１の辺を表す境界点集合に含まれる境界点のうち上記しきい値線より交点側に位置する境界点の中から上記しきい値直線に最も近い境界点をしきい値内近接点として検出すると共に、上記第２の辺を表す境界点集合に含まれる境界点のうち上記しきい値線より交点側に位置する境界点の中から上記しきい値直線に最も近い境界点をしきい値内近接点として検出する。そして、上記検出された第１の辺のしきい値内近接点と、上記検出された第２の辺のしきい値内近接点のうち、上記算出された交点候補点より外側にあるものはどれかを判定する。
このようにすることで、２次元コードの画像データの４辺において探索された境界点の集合に含まれるエラー点を、より一層確実に削除することが可能となる。
【００１３】
第４の構成は、２次元コードの歪みを矯正する際に以下のような処理を実行するものである。すなわち、先ず２次元コードの画像データの各頂点を結ぶ頂点フレームを生成する。次に、矯正対象の２次元コードの画像データにおける目標座標のピクセル値を算出するために、上記生成された頂点フレームの、上辺における上記目標座標に対応する点の座標値と、下辺における上記目標座標に対応する点の座標値と、左辺における上記目標座標に対応する点の座標値と、右辺における上記目標座標に対応する点の座標値をそれぞれ算出する。またそれと共に、上記２次元コードの画像データの、上辺における上記目標座標に対応する境界点の座標値と、下辺における上記目標座標に対応する境界点の座標値と、左辺における上記目標座標に対応する境界点の座標値と、右辺における上記目標座標に対応する境界点の座標値をそれぞれ算出する。さらに、上記頂点フレームにおいて上記目標座標に対応する目標座標値を算出する。そして、上記頂点フレームの上下左右の各辺について算出された上記目標座標に対応する点から、上記２次元コードの画像データの上下左右の各辺について算出された上記目標座標に対応する境界点へのベクトルを重み付け加算して、２次元コード歪みベクトルを算出し、上記頂点フレームについて算出された目標座標値に、上記算出された２次元コード歪みベクトルを加算して、上記２次元コードの画像デ

ータにおける上記目標座標に対応する目標座標値を算出する。
以上の処理を行うことで、曲線状歪みを有する２次元コードの画像データを長方形に矯正することが可能となる。
【発明の効果】
【００１４】
すなわちこの発明の一観点によれば、２次元コード画像が曲線状歪みを有する場合でも、この曲線状歪みを効果的に矯正して高精度のコード解読を可能にした２次元コード読取装置とそのプログラムを提供することができる。
【発明を実施するための最良の形態】
【００１５】
以下、図面を参照してこの発明に係わる実施形態を説明する。
図１は、この発明の一実施形態における２次元コード読取装置のハードウエア及びソフトウエアの構成を示すブロック図である。
本実施形態の２次元コード読取装置は、中央処理制御ユニット１と、カメラ２と、入力デバイス３と、表示デバイス４とを具備している。なお、２次元コード読取装置が携帯電話機に設けられる場合には、上記中央処理制御ユニット１、カメラ２、入力デバイス３及び表示デバイス４はいずれも携帯電話機が備える既存の構成を利用する。
【００１６】
中央処理制御ユニット１は、ＣＰＵ（Central Processing Unit）１１を備え、このＣＰＵ１１に対しバス１２を介してプログラムメモリ１３と、カメラインタフェース（カメラＩ／Ｆ）１４と、入出力インタフェース（入力Ｉ／Ｆ）１５と、データメモリ１６を接続したものとなっている。
【００１７】
カメラＩ／Ｆ１４にはカメラ２が接続される。カメラ２は、例えばＣＣＤ又はＣＭＯＳ等の半導体撮像素子を使用したものである。カメラＩ／Ｆ１４は、ＣＰＵ１１の制御の下で上記カメラ２を起動し、カメラ２により撮像された２次元コードの画像データを取り込む。
【００１８】
入出力Ｉ／Ｆ１５には入力デバイス３及び表示デバイス４が接続される。入力デバイス３は、ダイヤルキーと複数の機能キーとから構成される。なお、キーの構成は機構式のキーであっても、また表示デバイス４に表示されたソフトウエアキーであってもよい。表示デバイス４は、例えばＬＣＤ又はＥＬを使用したディスプレイからなる。入出力Ｉ／Ｆ１５は、上記入力デバイス３の操作データを取り込むと共に、ＣＰＵ１１の制御の下で表示データを上記表示デバイスに表示させる。
【００１９】
データメモリ１６には、携帯電話機の送受信に必要な電話帳や送受信履歴、送受信メール等の各種データを記憶する領域に加え、この発明に係わる２次元コード読取処理を実施するために必要なデータを一次記憶する領域を備える。２次元コード読取処理を実施するために必要なデータとしては、カメラ２により撮像された２次元コードの画像データやその二値化画像データ、処理過程で生成される種々データが含まれる。
【００２０】
プログラムメモリ１３には、この発明に係わる２次元コード読取処理を実施するために必要なアプリケーション・プログラムとして、二値化画像生成プログラム１３１と、ファインダパターン検出プログラム１３２と、画像回転処理プログラム１３３と、ファインダパターン境界検出プログラム１３４と、２次元コード境界線算出プログラム１３５と、２次元コード歪み矯正プログラム１３６が格納されている。
【００２１】
二値化画像生成プログラム１３１は、カメラ２により撮像された２次元コードの画像データをカメラＩ／Ｆ１４を介して取り込んでデータメモリ１６に一旦記憶させ、このデータメモリ１６から当該画像データを読み出して二値化画像データに変換し、この変換され

た二値化画像データをデータメモリ１６に記憶させる処理を、上記ＣＰＵ１１に実行させる。

【００２２】

ファインダパターン検出プログラム１３２は、上記データメモリ１６から上記二値化画像データを読み出し、この二値化画像データから２次元コードに含まれる複数のファインダパターン（例えばＱＲコードの場合には３個）を検出するための処理を、上記ＣＰＵ１１に実行させる。

【００２３】

画像回転処理プログラム１３３は、上記ファインダパターン検出処理により検出されたファインダパターンの位置に基づいて、上記２次元コードの画像が正立した状態となるように上記二値化画像データを回転させ、この回転処理後の二値化画像データをデータメモリ１６に記憶させる処理を、上記ＣＰＵ１１に実行させる。

【００２４】

ファインダパターン境界検出プログラム１３４は、上記回転処理後の二値化画像データをデータメモリ１６から読み出し、この読み出した二値化画像データから各ファインダパターンの黒枠の辺を表す境界点を検出してその座標値をデータメモリ１６に記憶させる処理を、上記ＣＰＵ１１に実行させる。

【００２５】

２次元コード境界線算出プログラム１３５は、上記検出された各ファインダパターンの黒枠の辺を表す境界点の座標値をデータメモリ１６から読み出し、この読み出した境界点の座標値に基づいて、２次元コードの輪郭を表す上下左右の各辺の形状を直線及び曲線で近似した近似線を生成する。そして、この生成された近似直線と近似曲線のうちいずれか妥当であるかを判定して、妥当と判定された近似線を上記２次元コードの上下左右の各辺の形状を表す境界線としてその座標値をデータメモリ１６に記憶させる処理を、上記ＣＰＵ１１に実行させる。

【００２６】

２次元コード歪み矯正プログラム１３６は、上記２次元コード境界線算出プログラム１３５により算出された上記２次元コード画像の上下左右の各辺の境界線を表す座標値をデータメモリ１６から読み出し、この読み出した上下左右の各辺の境界線をもとに、上記２次元コード画像の二値化画像データの歪みを矯正する処理を、上記ＣＰＵ１１に実行させる。

【００２７】

次に、以上のように構成された装置による２次元コードの読取処理動作を説明する。図２乃至図５はＣＰＵ１１によるその処理手順と処理内容を示すフローチャートである。

（１）２次元コードの画像データの取り込み

入力デバイス３においてユーザが撮像操作を行うと、ＣＰＵ１１はステップＳ１１により上記撮像操作を入出力Ｉ／Ｆ１５を介して検出する。そして、ステップＳ１２に移行し、カメラＩ／Ｆ１４を介してカメラ２を起動して、当該カメラ２により撮像された２次元コードの画像データをカメラＩ／Ｆ１４を介して取り込んでデータメモリ１６に記憶させる。

【００２８】

なお、上記二次元コードの画像データは複数のピクセルにより構成される。ここで、取り込んだ画像データの幅をwidth、高さをheightとすると、取り込んだ画像データはwidth×heightこのピクセルにより構成される。各ピクセルは、ＲＧＢカラーモデルにおいて色を表現するデータであり、フレームにおけるｘ座標値、ｙ座標値及びＲＧＢの各値によって構成され、ｘ座標値は０以上width未満の整数値、ｙ座標値は０以上height未満の整数値であり、ｘ軸は左から右方向へ、ｙ軸は上から下方向へそれぞれ向くものとする。またＲＧＢの各値はいずれも０以上２５５以下の整数値である。ここで、座標（x,y）におけるＲＧＢ各値が（r,g,b）であるとき、

ｐ（x,y）＝（r,g,b）

と表される。

【００２９】

（２）二値化画像データの生成処理

上記二次元コードの画像データが取り込まれると、ＣＰＵ１１はステップＳ２において二値化画像生成プログラム１３１を起動し、上記二次元コードの画像データのコントラストを強調するために、当該二次元コードの画像データを二値化画像データに変換する処理を以下のように実行する。

【００３０】

すなわち、先ずステップＳ２１により、上記データメモリ１６から二次元コードの画像データを読み出し、当該画像データの各ピクセルの輝度値を算出することによりグレースケール化する。例えば、

Grayscale（x,y）＝RGBtoLuminance（r,g,b）

によって輝度値を算出して、グレースケール画像を生成する。なお、輝度値は０以上２５５以下の値を持つものとする。

【００３１】

続いて、ステップＳ２２において、上記生成されたグレースケール画像の最大輝度値maxL、再使用輝度値minLをそれぞれ検出し、この検出された最大輝度値maxL、再使用輝度値minLを用いて、以下に示す(1)式によりコントラスト拡大画像を生成する。

【数１】

$$stretch(x,y)=\frac{255(grayScale(x,y)-minL)}{maxL-minL} \qquad \cdots(1)$$

【００３２】

さらに、ステップＳ２３において、例えばしきい値を１２８として下記(2)式に示す処理を実行し、二値化画像データを生成する。そして、この生成した二値化画像データをデータメモリ１６に記憶させる。

【数２】

$$binarize(x,y)=\begin{cases}1 & if\ 128\le stretch(x,y)\\ 0 & otherwise\end{cases} \qquad \cdots(2)$$

【００３３】

（３）ファインダパターンの検出処理

上記二値化画像データの生成が終了すると、ＣＰＵ１１は次にステップＳ３によりファインダパターン検出プログラム１３２を起動し、上記二値化画像データからＱＲコードに含まれる３個の位置パターン、つまりファインダパターンを検出するための処理を以下のように実行する。

【００３４】

すなわち、先ずステップＳ３１において、データメモリ１６から上記二値化画像データを読み出し、この二値化画像データに対しｙ軸の座標値を固定してｘ軸方向に走査する。そして、この走査により、例えば図６に示すように“白”、“黒”、“白”、“黒”、“白”のラインパターンを検出すると、各“白”と、各“黒”の長さをそれぞれ確認し、２個目の“黒”から６個目の“黒”までの長さの比、Ｂ１：Ｗ１：Ｂ２：Ｗ２：Ｂ３が１：１：３：１：１となっている箇所を検出し、この検出値をデータメモリ１６に保持させる。

【００３５】

具体的には、

Ｌ＝Ｂ１＋Ｗ１＋Ｂ２＋Ｗ２＋Ｂ３

として０以上のしきい値ｔに対して

【数３】

$$\left|\frac{B_1}{L}-\frac{1}{7}\right|\le\frac{1}{7}t,\left|\frac{W_1}{L}-\frac{1}{7}\right|\le\frac{1}{7}t,\left|\frac{B_2}{L}-\frac{3}{7}\right|\le\frac{3}{7}t,\left|\frac{W_2}{L}-\frac{1}{7}\right|\le\frac{1}{7}t,\left|\frac{B_3}{L}-\frac{1}{7}\right|\le\frac{1}{7}t \quad \cdots(3)$$

を満たす箇所を検出する。例えば、ｔ＝０．３等の値を用いる。

【００３６】

以下同様に、上記ステップＳ３１によるラインパターンの検出処理をｙ軸方向の全ての座標値について実行する。そして、全てのラインパターンの検出値が得られたことがステップＳ３２で確認されると、ステップＳ３３に移行して、上記検出された複数のラインパターンのうち互いに接するラインパターンをセグメント化し、このセグメントのサイズが大きい３個をＱＲコードのファインダパターンとして認識して、その座標値をデータメモリ１６に保存させる。

【００３７】

（４）画像回転処理

上記３個のファインダパターンの検出が終了すると、ＣＰＵ１１は続いてステップＳ４に移行して画像回線処理プログラム１３３を起動し、上記検出されたファインダパターンの位置をもとに上記二値化画像データを以下のように回転させる。

すなわち、先ずステップＳ４１において、上記検出された３個のファインダパターンＦ１，Ｆ２，Ｆ３の重心をそれぞれ算出する。そして、この算出された重心をＷ１＝（u1,v1）、Ｗ２＝（u2,v2）、Ｗ３＝（u3,v3）として、下記(4)式によりＤ１，Ｄ２，Ｄ３を算出する。

【数４】

$$\left.\begin{aligned}D_1&=(u2-u3)^2+(v2-v3)^2\\D_2&=(u3-u1)^2+(v3-v1)^2\\D_3&=(u1-u2)^2+(v1-v2)^2\end{aligned}\right\}\quad\cdots(4)$$

【００３８】

そして、上記Ｄ１，Ｄ２，Ｄ３の算出結果をもとに、ステップＳ４２において左上、右上、左下の各ファインダパターンを認識する。例えばＤ１が最大であればＦ１、Ｄ２が最大であればＦ２、Ｄ３が最大であればＦ３を左上ファインダパターンとして認識する。この処理は、右上と左上のファインダパターン間の距離が最大となることを利用している。図７にＤ１が最大となる場合の例を示す。

【００３９】

続いて、右上ファインダパターンと左下ファインダパターンを認識するが、Ｗ２とＷ３を通り直線に対しＷ１が右側にある場合にはＦ２を右上ファインダパターン、Ｆ３を左下ファインダパターンとし、そうでない場合にはＦ３を右上ファインダパターン、Ｆ２を左下ファインダパターンとする。

【００４０】

そうして左上、右上、左下の各ファインダパターンが認識されると、ＣＰＵ１１はステップＳ４３において、例えば図８に示すように左上ファインダパターンＦＰLTのｘ軸方向に右上ファインダパターンＦＰRTが位置するように、二次元コード画像の二値化画像データを回転させる。

【００４１】

（５）ファインダパターン境界の検出処理

次にＣＰＵ１１は、ステップＳ５によりファインダパターン境界検出プログラム１３４

を起動し、上記回転処理後の二値化画像データから３個のファインダパターンの黒枠の辺を表す境界点を検出する処理を以下のように実行する。

すなわち、ＣＰＵ１１は先ずステップＳ５１において、例えば図９に示すように黒枠のセグメントＳを抽出し、黒枠の４角の座標を算出する。４角の座標は、セグメントＳに含まれる座標のうち、－ｘ－ｙが最大となる座標を左上の角（cx1,cy1）、ｘ－ｙが最大となる座標を右上の角（cx2,cy2）、ｙ－ｘが最大となる座標を左下の角（cx3,cy3）、ｘ－ｙが最大となる座標を右下の角（cx4,cy4）となるようにそれぞれ設定する。

【００４２】

ＣＰＵ１１は次にステップＳ５２において、上記黒枠のセグメントの上下左右の各辺を検出する。例えば、上辺FinderTopは、

【数５】

$$FinderTop = \{(x,y) \mid cx_1 \le x \wedge x \le cx_2 \wedge y = \min(\{y' \mid (x,y') \in S\})\} \quad \cdots (5)$$

として検出される。

【００４３】

これは、図１０に示すように黒枠のセグメントＳ内において左上の角と右上の角との間のｘについて、それぞれｙが最小となる点の集合を表す。同様に、下辺FinderBottom、左辺FinderLeft、右辺FinderRightは、

【数６】

$$\left.\begin{aligned} FinderBottom &= \{(x,y) \mid cx_3 \le x \wedge x \le cx_4 \wedge y = \max(\{y' \mid (x,y') \in S\})\} \\ FinderLeft &= \{(x,y) \mid cy_1 \le y \wedge y \le cy_3 \wedge x = \min(\{x' \mid (x',y) \in S\})\} \\ FinderRight &= \{(x,y) \mid cy_2 \le y \wedge y \le cy_4 \wedge x = \max(\{x' \mid (x',y) \in S\})\} \end{aligned}\right\} \cdots (6)$$

として検出される。

【００４４】

（６）２次元コード境界線の算出処理

ＣＰＵ１１は、次にステップＳ６に移行して２次元コード境界線算出プログラム１３５を起動し、上記検出された３個のファインダパターンの黒枠の上下左右各辺の座標値に基づいて、２次元コード画像の輪郭を表す上下左右各辺に含まれる座標をそれぞれ検出し、その近似線を上下左右各辺の境界線として算出する処理を実行する。

【００４５】

例えば、２次元コード画像の上辺の境界点座標を検出する場合、ＣＰＵ１１はステップＳ６１～Ｓ６８により以下の処理を行う。すなわち、上辺の左側には左上ファインダパターンＦＰLT、右側には右上ファインダパターンＦＰRTが位置する。それぞれのファインダパターンＦＰLT，ＦＰRTの上辺の境界点座標は、先に述べたファインダパターン境界検出処理により既に検出されているので、その検出結果を利用してファインダパターンＦＰLT，ＦＰRT間に位置する２次元コード画像の上辺上の境界点の集合をステップＳ６２により検出する。

【００４６】

ここで、左側のファインダパターンＦＰLTの上辺の境界点の集合をLeftFinderEdge、右側のファインダパターンＦＰRTの上辺の境界点の集合をRightFinderEdgeとし、LeftFinderEdgeの右端の境界点を（LeftEndX，LeftEndY）、RightFinderEdgeの左端の境界点を（RightStartX，RightStartY）とする。また、LeftFinderEdgeに含まれる境界点の重心を（LeftCenterX，LeftCenterY）、RightFinderEdgeに含まれる座標点の重心を（RightCenterX，RightCenterY）とする。また、LeftFinderEdgeから最小二乗法によって近似した直線を

y=alx+bl、RightFinderEdgeから最小二乗法によって近似した直線をy=arx+brとする。この最小二乗法によって得られた直線から、あるｘ座標における上辺の傾きedgeSlope(x)は、

【数７】

$$edgeSlope(x)=\frac{a_l\left|RightCenterX-x\right|+a_r\left|x-LeftCenterY\right|}{\left|RightCenterX-LeftCenterY\right|} \quad \cdots(7)$$

と推測できる。

【００４７】

次にＣＰＵ１１は、データメモリ１６から読み出した二値化画像データにおいて、LeftEndXとRightStartXとの間の２次元コード画像の上辺を検出する。その検出処理は例えば図１０に示すように、検出された上辺上の点P=(px,py)に対してｘ座標がpx+k（初期条件ではk=1）となる上辺上の点を検出するためにqy及びryを設定する。そして、qyからryまで下方向に“白”から“黒”に変わる最初の境界点を探索し、見つかった境界点P′=（px+k,py′）を新たな上辺上の境界点として検出する。ここで、qy及びryは、定数cq、cr,dq、drを用いて

【数８】

$$\left.\begin{aligned} q_y &= p_y+\min\left(c_q,\left(edgeSlope\left(p_x+\frac{k}{2}\right)-d_q\right)k\right) \\ r_y &= p_y+\max\left(c_r,\left(edgeSlope\left(p_x+\frac{k}{2}\right)+d_r\right)k\right) \end{aligned}\right\} \cdots(8)$$

として求める。なお、定数については、例えばcq=5、cr=0、dq=0.1、dr=0.1等の値を用いる。

【００４８】

一方、以上の処理により新たな境界点が見つからない場合には、ｋの値を１ずつ増やして探索し、見つかった場合にはP′に対してk=1から同様の処理を行うことで次の境界点を探索する。P=（LeftEndX,LeftEndY）から探索を開始し、px+k=RightStartXとなれば探索終了である。以上の処理によって、左上ファインダパターンＦＰLTと右上ファインダパターンＦＰRTとの間に位置する２次元コードの上辺上の境界点の集合MiddleEdgeが検出される。

【００４９】

以上のステップＳ６２の処理により検出された２次元コードの上辺上の境界点集合MiddleEdgeには、本来の上辺より２次元コード画像の内側の点も含まれる。そこでＣＰＵ１１は、続いて２次元コード画像内の点の除去を行う。このエラー点の除去方法には、上辺の傾きから推測される範囲に含まれない点をエラーとして除去する第１の方法と、互いに直交する２つの辺の交点を求め、この交点より外側の範囲に位置する境界点をエラー点として削除する第２の方法とがある。ここでは、先ず第１の方法について説明し、第２の方法は後に詳しく説明する。

【００５０】

第１の方法は以下のように行われる。すなわち、ＣＰＵ１１はエラー点の除去処理を右から左に向かって行う。ここで、点P=(px,py)に対し、MiddleEdgeに含まれ、Pより左に位置する最初の点をP′=(px′,py′)とする。このとき線分PP′の傾きは、

【数９】

$$\frac{p_y'-p_y}{p_x'-p_x} \quad \cdots(9)$$

と表される。

【００５１】

ここで、あるｘにおける上辺の傾きより定数ct（例えば0.1）だけ小さい傾きを返す関数edgeMinSlope(x)を、

【数１０】

$$edgeMinSlope(x) = edgeSlope(x) - c_t \quad \cdots(10)$$

と定義する。

【００５２】

ここで、Ｐ′がエラーであるか否かを、

【数１１】

$$edgeMinSlope\left(\frac{p_x+p_x'}{2}\right) < \frac{p_y'-p_y}{p_x'-p_x} \quad \cdots(11)$$

によって判定する。この式は、

【数１２】

$$p_y' < p_y - (p_x - p_x')edgeMinSlope\left(\frac{p_x+p_x'}{2}\right) \quad \cdots(12)$$

とも表現される。

【００５３】

この判定処理は、図１１に示すように、線分PQの傾きを推定される上辺の傾きとし、線分PRの傾きをPRの傾きよりctだけ小さい傾きであるとすると、P1′やP2′のようにRより上に位置する点はエラーではないが、P3′のようにRより下に位置する点はエラーとして検出される。ただし、図１１においてQ、R、P1′、P2′、P3′のｘ座標値は全てpx′であるとする。上記手順によって、MiddleEdgeからエラー点が除去される。

【００５４】

以上の手順によって、LeftFinderEdge、MiddleEdge、RightFinderEdgeによって２次元コード画像の上辺の境界点集合が検出される。ここで上辺の点集合TopEdgeは

【数１３】

$$TopEdge = LeftFinderEdge \cup MiddleEdge \cup RightFinderEdge \quad \cdots(13)$$

と表される。

【００５５】

次にＣＰＵ１１は、ステップＳ６４，Ｓ６５において、上記上辺の点集合TopEdge をもとにその近似線を算出する。TopEdgeの近似線は、近似直線或いは近似曲線によって表現する。TopEdgeに含まれる点列を

(x1,y1),(x2,y2),…,(xn,yn)

とすると、最小二乗法による近似直線y=ax+bは

【数１４】

$$\left.\begin{aligned} a &= \frac{n\sum_{k=1}^{n} x_k y_k - \sum_{k=1}^{n} x_k \sum_{k=1}^{n} y_k}{n\sum_{k=1}^{n} x_k{}^2 - \left(\sum_{k=1}^{n} x_k\right)^2} \\ b &= \frac{\sum_{k=1}^{n} x_k{}^2 \sum_{k=1}^{n} y_k - \sum_{k=1}^{n} x_k y_k \sum_{k=1}^{n} x_k}{n\sum_{k=1}^{n} x_k{}^2 - \left(\sum_{k=1}^{n} x_k\right)^2} \end{aligned}\right\} \cdots (14)$$

によって求まる。

【００５６】

一方、近似曲線は最小二乗法による円への近似により求める。最小二乗法による円への近似では、(a,b)を中心とする半径Rの円$(x-a)^2+(y-b)^2=R$を求めるために

【数１５】

$$F(a,b,R)=\sum_{i=1}^{n}\left((x_i-a)^2+(y_i-b)^2-R^2\right)^2=\sum_{i=1}^{n}\left(z_i+Bx_i+Cy_i+D\right)^2 \cdots (15)$$

の最小化を行う。

【００５７】

ただし、z_i,B,C,Dはそれぞれ

【数１６】

$$z_i=x_i^2+y_i^2, B=-2a, C=-2b, D=a^2+b^2-R^2 \cdots (16)$$

とする。

【００５８】

F(a,b,R)をB、C、Dについてそれぞれ偏微分すると、

【数１７】

$$\left.\begin{aligned} \left(\sum_{i=1}^{n} x_i^2\right)B+\left(\sum_{i=1}^{n} x_i y_i\right)C+\left(\sum_{i=1}^{n} x_i\right)D &= -\left(\sum_{i=1}^{n} x_i z_i\right) \\ \left(\sum_{i=1}^{n} x_i y_i\right)B+\left(\sum_{i=1}^{n} y_i^2\right)C+\left(\sum_{i=1}^{n} y_i\right)D &= -\left(\sum_{i=1}^{n} y_i z_i\right) \\ \left(\sum_{i=1}^{n} x_i\right)B+\left(\sum_{i=1}^{n} y_i\right)C+nD &= -\left(\sum_{i=1}^{n} z_i\right) \end{aligned}\right\} \cdots (17)$$

となる。この連立方程式をCholesky分解により解き、B、C、Dを得て、

【数１８】

$$a=-\frac{B}{2}, b=-\frac{C}{2}, R=\sqrt{\frac{B^2+C^2}{4}-D} \cdots (18)$$

によりa、b、Rを得る。

【００５９】
次にＣＰＵ１１は、ステップＳ６６において近似直線と近似曲線とを比較してどちらが妥当であるかを判定する。この判定は、近似直線から各(x_i,y_i)への距離の和と、近似曲線から各(x_i,y_i)への距離の和を比較し、距離の和が小さい方の近似線が妥当であると判定することにより正確に行うことができる。なお、計算量を低減したい場合には、(x_1,y_1)、(x_M,y_M)、(x_n,y_n)への距離の和によって判定しても良い。ただし、mは
【数１９】

$$m=\left\lfloor\frac{n}{2}\right\rfloor \quad \cdots (19)$$

によって求まる整数であり、(x_M,y_M)は検出された上辺の点集合の中央の点である。以上の上辺の近似線算出の手順を図１２のステップＳ１２１～Ｓ１２６に示す。
【００６０】
以上の手順によって上辺の境界線が算出される。同様に、図１３に示すように左辺についても、xy座標系を左へ９０度回転させて考える（上方向をx軸方向、右方向をy軸方向とする）ことで、同じ手順によって境界線が算出される。また、下辺については、xy座標系のy方向を逆転させることで、２次元コード画像の下辺の点集合を算出できる。ただし、ＱＲコードには右下にファインダパターンはないため、RightFinderEdgeは存在しない。そこで、edgeSlope(x)は、(LeftCenterX,LeftCenterY)から(px′,py′)へ引いた線分の傾き
【数２０】

$$edgeSlope(x)=\frac{p_y'-LeftCenterY}{p_x'-LeftCenterX} \quad \cdots (20)$$

によって算出する。
【００６１】
また、RightFinderEdgeが存在しないため、探索は２次元コード画像の端まで行うものとし、そうして検出した点の集合をMiddleBottomEdgeとする。また右辺についても、xy座標系を右へ９０度回転させて考える（下方向をx軸方向、左方向をy軸方向とする）ことで２次元コード画像の下辺の点集合を検出できる。しかし、この場合も上辺におけるRightFinderEdgeが存在しないので、edgeSlope(x)は下辺と同様に
【数２１】

$$edgeSlope(x)=\frac{p_y'-LeftCenterY}{p_x'-LeftCenterX} \quad \cdots (21)$$

によって算出し、こちらも２次元コード画像の端まで探索を行うものとし、そうして検出した点の集合をMiddleRightEdgeとする。
【００６２】
次に、２次元コード画像の右下の角の座標RightBottomを検出する。初期のxy座標系（右方向がx軸方向、下方向がy軸方向）におけるMiddleBottomEdgeを
【数２２】

$$MiddleBottomEdge=\{b_{k=1,2..,m}\}=\{(bx_1,by_1),(bx_2,by_2),...,(bx_m,by_m)\} \quad \cdots (22)$$

とし、同様に初期のxy座標系におけるMiddleRightEdgeを

【数２３】

$$MiddleRightEdge = \{r_{k=1,2,..,n}\} = \{(rx_1, ry_1), (rx_2, ry_2), ..., (rx_n, ry_n)\} \cdots (23)$$

とする。このときbx$_1$<bx$_2$<...<bx$_M$及びry$_1$<ry$_2$<...<ry$_N$となる。また、左下ファインダパターンＦＰLBは右上ファインダパターンＦＰRTより左下に位置すると考えられるので、bx$_1$<rx$_1$かつry$_1$<by$_1$となる。

【００６３】

またＣＰＵ１１は、RightBottomは２次元コードにおいてf(x,y)=x+yが最大となる点だと仮定し、MiddleBottomEdgeとMiddleRightEdgeからRightBottomの近くに位置する点を探索する。ただし、MiddleBottomEdgeはRightBottomより右側の点を、MiddleRightEdgeはRightBottomより下側の点を含んでいう可能性があるので、これらのエラー点を除去する必要がある。

【００６４】

このエラー点の除去は、先に述べた第２の方法により以下のように行われる。図４はその処理の一部である交点算出処理の手順を示すフローチャートである。

すなわち、ＣＰＵ１１は、ステップＳ６３１～Ｓ６３５において、例えば図１４に示すようにMiddleBottomEdgeに含まれる各点(bxi,byi)から傾き－１の閾値直線、つまり

y＝－(x-bxi)＋bxi

を引き、各直線の上方向に位置し、bxi≦rxjでかつy座標の最も大きいMiddleRightEdge内の点を示すjの値を算出する関数f(i)を

【数２４】

$$f(i) = \max\left(\left\{j \middle| bx_i \le rx_j \wedge ry_j \le -(rx_j - bx_i) + by_i\right\}\right) \cdots (24)$$

とする。そして、j=f(i)としたとき(bxi, byi)および(rxj, ryj)からRightBottomを推定する関数InferRightBottom(i)を

【数２５】

$$InferRightBottom(i) = (rx_j, by_i) \cdots (25)$$

とする。

【００６５】

ここでInferRightBottom(i)と比較してb$_{i+1}$が右側にあるか、r$_{j+1}$が下側にあるような最小のiをBottomEndとすると、BottomEndは

【数２６】

$$BottomEnd = \min\left(\left\{i \middle| \left(rx_j \le bx_{i+1} \vee by_i \le ry_{j+1}\right)\right\}\right) \cdots (26)$$

となる。上記BottomEndを用いてRightBottomは、

【数２７】

$$InferRightBottom(BottomEnd) \cdots (27)$$

によって検出される。

【００６６】

次にＣＰＵ１１は、MiddleBottomEdgeからRightBottomより右側に位置する点（x座標値の大きい点）を除去する。また、MiddleRightEdgeからRightBottomより下側に位置する点（y座標値の大きい点）を除去する。さらに、MiddleBottomEdgeについて再びxy座標系のy方向を逆転させ、上辺における処理と同様に図１１に示したような２次元コード画像内のエラー点の除去を行う。MiddleRightEdgeについては、xy座標系を右へ９０度回転させて、同様に２次元コード画像内のエラー点の除去を行う。

【００６７】

さらにＣＰＵ１１は、左下ファインダパターンＦＰLBの下辺をLeftFinderBottomEdge、右上ファインダパターンＦＰRTの右辺をTopFinderRightEdgeとして、２次元コード画像の下辺の点集合BottomEdge及び右辺の点集合RightEdgeを、

【数２８】

$$\left.\begin{aligned} BottomEdge &= LeftFinderBottomEdge \cup MiddleBottomEdge \\ RightEdge &= TopFinderRightEdge \cup MiddleRightEdge \end{aligned}\right\} (28)$$

とする。

【００６８】

ＣＰＵ１１は、以上の手順によって求めた下辺の点集合BottomEdgeから、上辺において近似線を算出した手順と同様の手順によって近似線を算出し、再び元のxy座標系に戻すことで下辺の近似線を算出する。また、右辺の点集合からも同様に近似線が算出され、元のxy座標系に戻すことで右辺の近似線が算出される。

【００６９】

（７）２次元コード画像の歪みの矯正

以上の処理により２次元コード画像の上下左右各辺についての境界線が算出されると、続いてＣＰＵ１１は図５に示すステップＳ７において２次元コード歪み矯正プログラム１３６を起動し、上記算出された２次元コード画像の上下左右各辺の境界線を利用して２次元コード画像の歪みの矯正を次のように行う。

【００７０】

すなわち、２次元コード画像の歪みの矯正においては、２次元コード画像をその幅codeWidth、高さcodeHeightについてそれぞれ矯正するものとし、矯正後の２次元コードにおける目標座標(x,y)のピクセル値を求めることによって矯正後の２次元コードを生成する。

【００７１】

ＣＰＵ１１は、先ずステップＳ７１において、２次元コードの４角の座標を算出する。２次元コードの４角の座標は、上辺と左辺の境界線から左上角の点LeftTopを、上辺と右辺の境界線から右上角の点RightTopを、下辺と左辺の境界線から左下角の点LeftBottomを、左辺と右辺の境界線から右下角の点RightBottomを算出する。

【００７２】

次に、LeftTop、RightTop、RightBottom、LeftBottomの４点を結ぶ四角形BaseFrameを作成する。そして、

【数２９】

$$\left.\begin{aligned} TopWeight &= \frac{codeHeight - 1 - y}{codeHeight - 1} \\ LeftWeight &= \frac{codeWidth - 1 - x}{codeWidth - 1} \end{aligned}\right\} \cdots (29)$$

とした上で、上記作成されたBaseFrameにおいて、図１５に示すように、LeftTopからの距

離とRightTopからの距離の比が(1-LeftWeight)LeftWeightとなる上辺上の点をBaseTop(LeftWeight)とし、LeftBottomからの距離とRightBottomからの距離の比が(1-LeftWeight)LeftWeightとなる点をBaseBottom(LeftWeight)とし、LeftrTopからの距離とLeftBottomからの距離の比が(1-TopWeight)TopWeightとなる点をBaseLeft(TopWeight)とし、RightTopからの距離とRightBottomからの距離の比が(1-TopWeight)TopWeightとなる点をBaseRight(TopWeight)とする。

【００７３】

続いてＣＰＵ１１は、ステップＳ７２，Ｓ７３により、BaseFrameにおいて目標座標に対応する座標BaseCross(TopWeight,LeftWeight)を求める。ただし、図１５に示すようにBaseTop(LeftWeight)とBaseBottom(LeftWeight)とを結ぶ線分と、BaseLeft(TopWeight)とBaseRight(TopWeight)とを結ぶ線分の交点をBaseCross(TopWeight,LeftWeight)とする。

BaseCross(TopWeight,LeftWeight)は、

【数３０】

$$\begin{aligned} &BaseCross(TopWeight, LeftWeight) \\ &= TopWeight \cdot BaseTop + (1 - TopWeight)BaseBottom \\ &= LeftWeight \cdot BaseLeft + (1 - LeftWeight)BaseRight \end{aligned} \quad \cdots (30)$$

を満たす。

【００７４】

またＣＰＵ１１は、ステップＳ７４において、図１６に示すようにBaseTopとx座標値が同一となる上辺上の点をTopPoint(LeftWeight)とし、BaseBottomとx座標値が同一となる下辺上の点をBottomPoint(LeftWeight)とし、LeftBaseとy座標値が同一となる左辺上の点をLeftPoint(TopWeight)とし、RightBaseとy座標値が同一となる右辺上の点をRightPoint(TbpWeight)とする。

【００７５】

そして、ステップＳ７５において、図１６に示すように２次元コード画像における上辺の歪みDistortionTop(LeftWeight)、下辺の歪みDistortionBottom(LeftWeight)、左辺の歪みDistortionLeft(TopWeight)、右辺の歪みDistortionRightRight(TopWieight)をそれぞれ

【数３１】

$$\left.\begin{aligned} &DistortionTop(LeftWeight) = TopPoint - BaseTop \\ &DistortionBottom(LeftWeight) = BottomPoint - BaseBottom \\ &DistortionLeft(TopWeight) = LeftPoint - BaseLeft \\ &DistortionRight(TopWeight) = RightPoint - BaseRight \end{aligned}\right\} \cdots (31)$$

とし、歪みベクトルDistortionVector(TopWeight,LeftWeight)を

【数３２】

$$\begin{aligned} &DistortionVector(TopWeight, LeftWeight) \\ &= TopWeight \cdot DistortionTop(LeftWeight) + (1 - TopWeight)DistortionBottom(LeftWeight) \\ &+ LeftWeight \cdot DistortionLeft(TopWeight) + (1 - LeftWeight)DistortionRight(TopWeight) \end{aligned} \quad \cdots (32)$$

とする。

【００７６】
次にＣＰＵ１１は、ステップＳ７６において、図１７に示すようにBaseCross(TopWeight, LeftWeight)にDistortionVbctor(TopWeight,LeftWeight)を加えた点を算出する関数CrossPoint(TopWeight,LeftWeight)を
【数３３】

$$CrossPoint(TopWeight, LeftWeight) = BaseCross(TopWeight, LeftWeight) + DistortionVector(TopWeight, LeftWeight) \quad \cdots (33)$$

とする。なお、CrossPoint(TopWeight,LeftWeight)は、
【数３４】

$$\begin{aligned} &CrossPoint(TopWeight, LeftWeight) \\ &= TopWeight \cdot TopPoint + (1 - TopWeight)BottomPoint \\ &\quad + LeftWeight(LeftPoint - BaseLeft) + (1 - LeftWeight)(RightPoint - BaseRight) \\ &= LeftWeight \cdot LeftPoint + (1 - LeftWeight)RightPoint \\ &\quad + TopWeight(TopPoint - BaseTop) + (1 - TopWeight)(BottomPoint - BaseBottom) \end{aligned} \quad \cdots (34)$$

を満たし、DistortionVector(TopWeight, LeftWeight)を算出せずに求めることもできる。
【００７７】
ＣＰＵ１１は、以上のように定義したCrossPoint(TopWeight,LeftWeight)を用いて、２次元コード画像の歪みの矯正を行う。TopWeight及びLeftWeightは上記にて示した定義の通りである。したがって、矯正後の２次元コード画像における座標(x,y)のピクセル値には、矯正前の２次元コード画像における
【数３５】

$$CrossPoint\left(\frac{codeHeight - 1 - y}{codeHeight - 1}, \frac{codeWidth - 1 - x}{codeWidth - 1}\right) \quad \cdots (35)$$

のピクセル値と同一の値を持たせる。
【００７８】
以上の処理により、図１８（ａ），（ｂ）に示すような糸巻き型又は樽型の曲線歪みを有する２次元コード画像を長方形に矯正することが可能となる。また、codeWidthとcodeHeightを同一の値に設定することにより、正方形への矯正も可能となる。
【００７９】
そしてＣＰＵ１１は、以上の処理によって矯正された２次元コード画像を、ステップＳ８によりコード解析し、このコード解析されたコードをデータメモリ１６に記憶させる。このデータメモリ１６に記憶されたコードは、例えば入力デバイス３においてユーザが２次元コードの送信操作を行った場合に、データメモリ１６から読み出されてＷｅｂサーバ等に送信される。
【００８０】
以上説明したようにこの実施形態では、カメラ２により撮像された２次元コードの画像データを二値化画像に変換した後、当該二値化画像から２次元コードのファインダパター

ンを検出し、この検出されたファインダパターンの上下左右各辺の座標値を検出する。そして、この検出されたファインダパターンの上下左右各辺の座標値をもとに、上記２次元コードの画像データの上下左右各辺の形状を直線及び曲線で近似した近似線を生成し、この生成された近似線をもとに上記２次元コードの画像データの歪みを矯正して、この歪みが矯正された２次元コードの画像データをもとに上記２次元コードを解読するようにしている。

【００８１】

したがって、撮像した２次元コードの画像が例えば糸巻き型又は樽型の曲線状歪みを有していても、２次元コードの画像データの上下左右各辺の形状が直線及び曲線により近似され、その近似線をもとに上記２次元コードの画像データの歪みが矯正される。このため、上記糸巻き型又は樽型の曲線状歪みを有する２次元コード画像に対しても高精度のコード解読を行うことが可能となる。

【００８２】

なお、この発明は上記実施形態に限定されるものではない。例えば、前記実施形態では二次元コード読取装置を携帯電話機に内蔵した場合を例にとって説明したが、二次元コード読取装置をテレビジョン受信機やビデオ記録再生装置のリモートコントローラに内蔵するようにしてもよく、またパーソナル・コンピュータに内蔵するようにしてもよい。また、前記実施形態では二次元コード読取装置がカメラを内蔵する場合を例にとって説明したが、二次元コード読取装置とは別に設けられたディジタルカメラにより２次元コードを撮像し、その画像データを信号ケーブル又は記録媒体を介して二次元コード読取装置に取り込むようにしてもよい。

【００８３】

その他、二値化画像生成処理、ファインダパターン検出処理、画像回転処理、ファインダパターン境界検出処理、２次元コード境界線算出処理及び２次元コード歪み矯正処理の各処理手順と処理内容についても、この発明の要旨を逸脱しない範囲で種々変形して実施できる。

【００８４】

要するにこの発明は、上記実施形態そのままに限定されるものではなく、実施段階ではその要旨を逸脱しない範囲で構成要素を変形して具体化できる。また、上記実施形態に開示されている複数の構成要素の適宜な組み合せにより種々の発明を形成できる。例えば、実施形態に示される全構成要素から幾つかの構成要素を削除してもよい。さらに、異なる実施形態に亘る構成要素を適宜組み合せてもよい。

【図面の簡単な説明】

【００８５】

【図１】この発明の一実施形態に係わる二次元コード読取装置のハードウエア及びソフトウエアの構成を示すブロック図である。

【図２】図１に示した二次元コード読取装置による二次元コード読取制御のうち、撮像制御から画像回転処理までの処理手順と処理内容を示すフローチャートである。

【図３】図１に示した二次元コード読取装置による二次元コード読取制御のうち、ファインダパターン境界検出処理から２次元コード境界線算出処理までの処理手順と処理内容を示すフローチャートである。

【図４】図３に示した２次元コード境界線算出処理過程におけるエラー点除去のための交点算出処理手順とその内容を示すフローチャートである。

【図５】図１に示した二次元コード読取装置による二次元コード読取制御のうち、２次元コード歪み矯正処理の手順と内容を示すフローチャートである。

【図６】ファインダパターン検出処理の説明に用いる図である。

【図７】ファインダパターン検出処理の説明に用いる図である。

【図８】２次元コード画像の回転処理の説明に用いる図である。

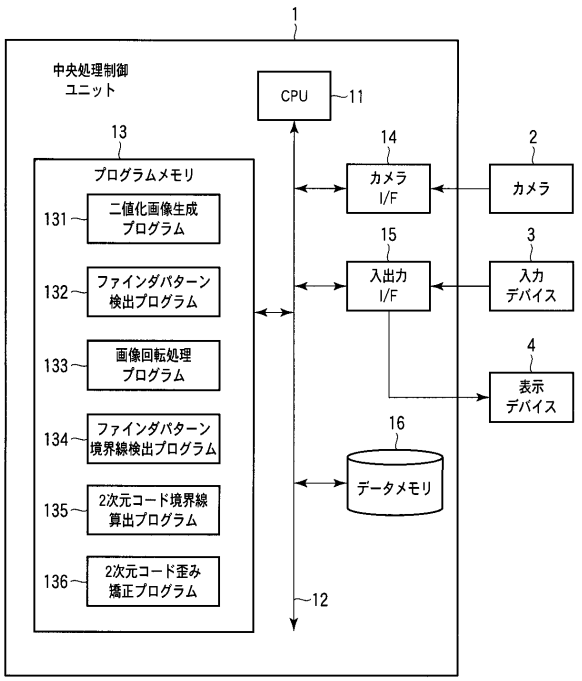
【図９】ファインダパターンの上下左右各辺の座標値を検出する処理に用いる図である。

【図１０】ファインダパターンの上下左右各辺の座標値を検出する処理に用いる図である

。
【図１１】２次元コード境界線を算出する処理において、エラー点を除去する第１の処理を説明するための図である。
【図１２】２次元コード境界線を算出する処理において、２次元コード画像の上辺を直線及び曲線で近似するときの手順を説明するための図である。
【図１３】２次元コード境界線を算出する処理を説明するための図である。
【図１４】２次元コード境界線を算出する処理において、エラー点を除去する第２の処理を説明するための図である。
【図１５】２次元コード画像の歪み矯正処理の説明に用いる図である。
【図１６】２次元コード画像の歪み矯正処理の説明に用いる図である。
【図１７】２次元コード画像の歪み矯正処理の説明に用いる図である。
【図１８】撮像された２次元コード画像に発生する糸巻き型及び樽型歪みの一例を示す図である。
【図１９】従来の２次元コード読取装置を説明するための図である。
【符号の説明】
【００８６】
１…中央処理制御ユニット、２…カメラ、３…入力デバイス、４…表示デバイス、１１…ＣＰＵ、１２…バス、１３…プログラムメモリ、１４…カメラＩ／Ｆ、１５…入出力Ｉ／Ｆ、１６…データメモリ、１３１…二値化画像生成プログラム、１３２…ファインダパターン検出プログラム、１３３…画像回転処理プログラム、１３４…ファインダパターン境界検出プログラム、１３５…２次元コード境界線算出プログラム、１３６…２次元コード歪み矯正プログラム。

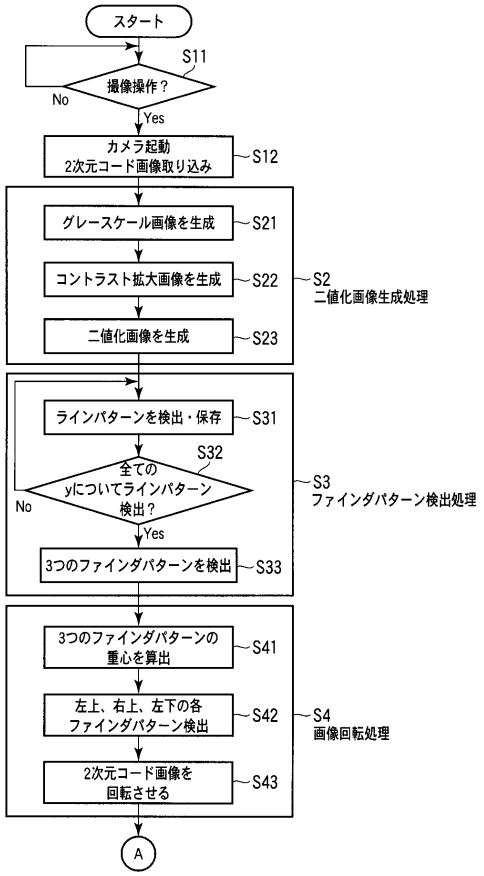
【図１】

図１



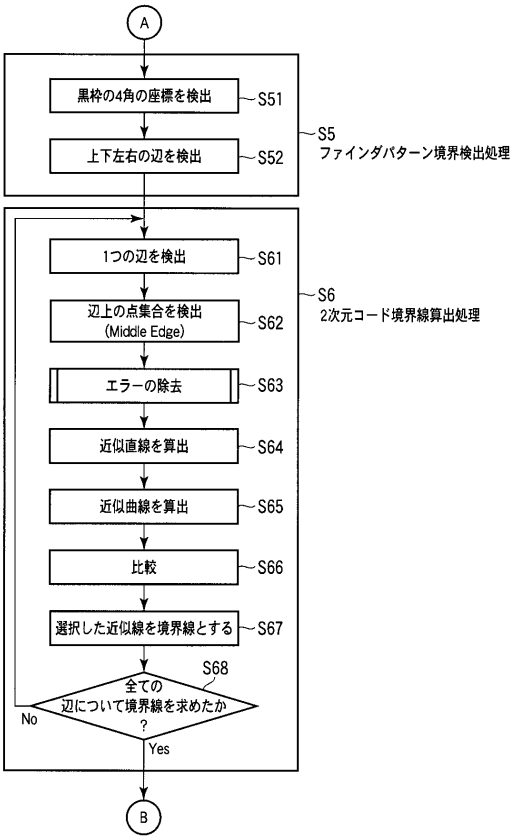
【図２】

図２



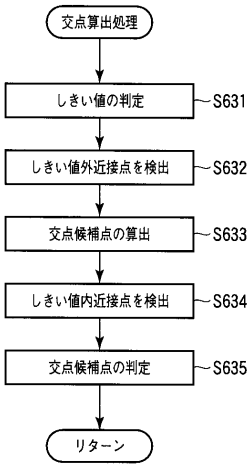
【図３】

図３



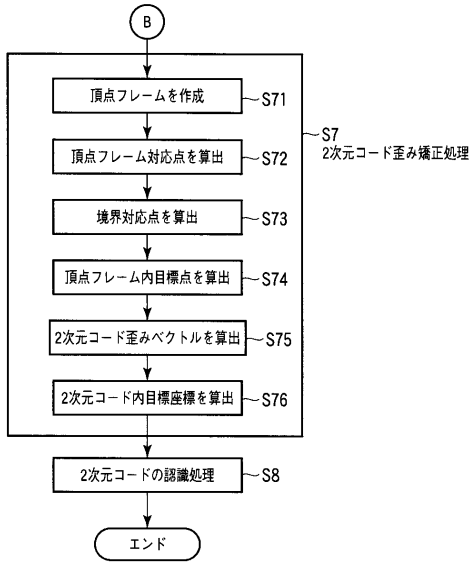
【図４】

図４



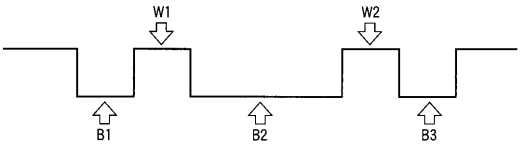
【図５】

図５



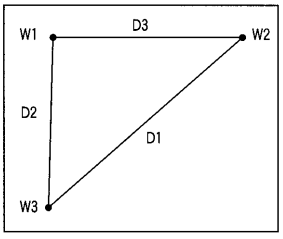
【図６】

図６



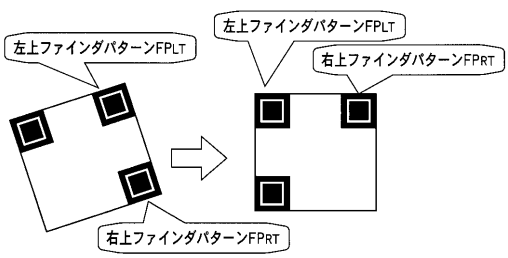
【図７】

図７



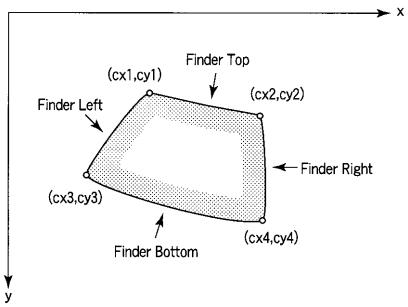
【図８】

図８



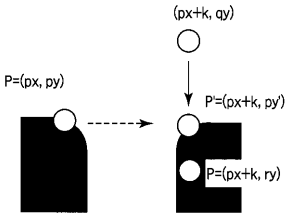
【図９】

図 9



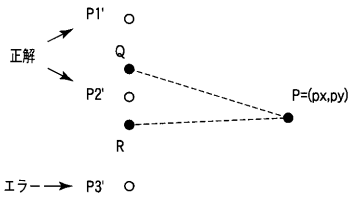
【図１０】

図 10



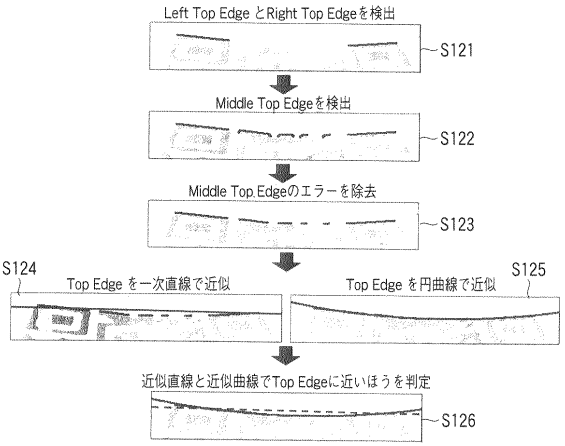
【図１１】

図 11



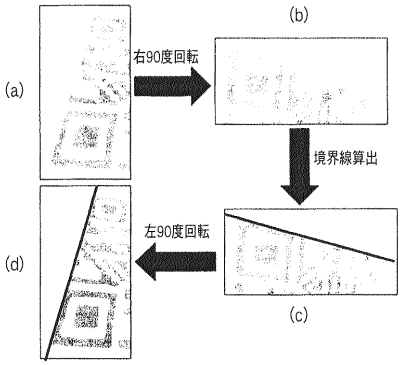
【図１２】

図 12



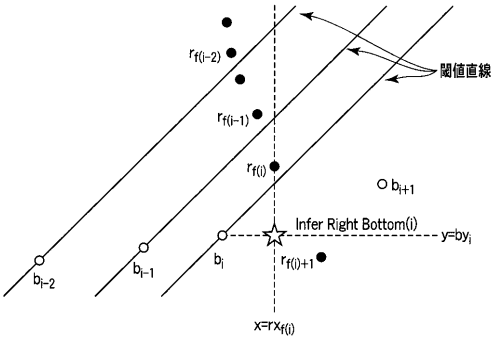
【図１３】

図 13



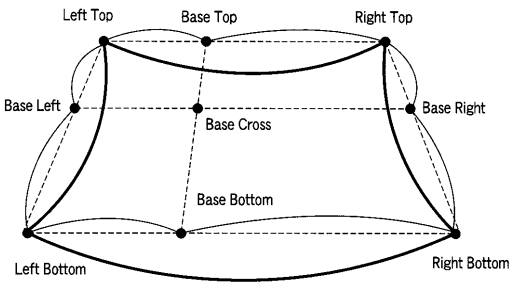
【図１４】

図 14



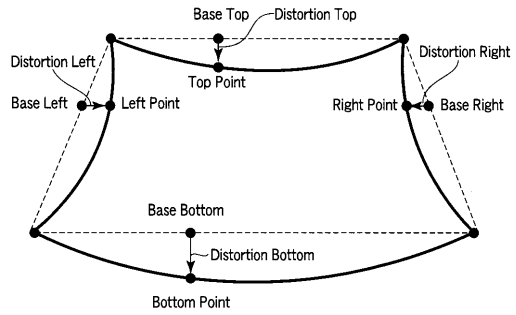
【図１５】

図 15



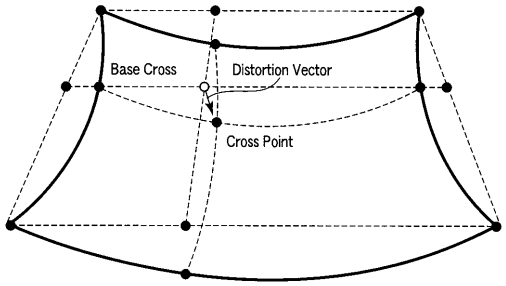
【図１６】

図16



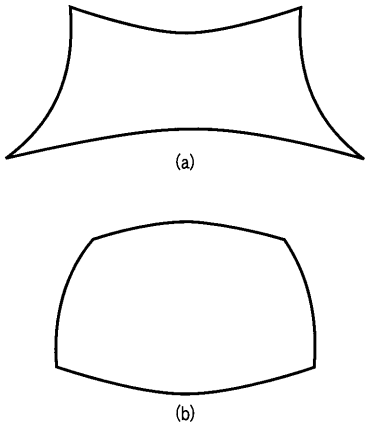
【図１７】

図17



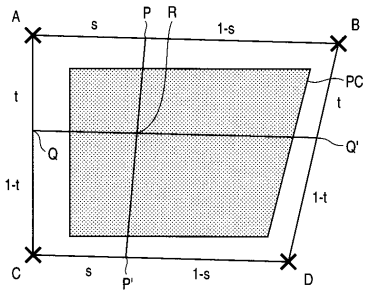
【図１８】

図18



【図１９】

図19



フロントページの続き

(74)代理人 100103034
弁理士 野河 信久
(72)発明者 原 謙治
東京都千代田区大手町二丁目３番１号 日本電信電話株式会社内
(72)発明者 前田 篤彦
東京都千代田区大手町二丁目３番１号 日本電信電話株式会社内
(72)発明者 稲垣 博人
東京都千代田区大手町二丁目３番１号 日本電信電話株式会社内

Ｆターム(参考) 5B035 BB08
5B057 BA02 CA08 CA12 CA16 CD12 CE09 DA08 DC07 DC16
5B072 CC21 FF00
5C076 AA23 AA31
5C077 PP20 PP59 TT09 TT10