16.06.2015

Dokumentation

QRTrace

Thomas Schrotter, Markus Schrotter, Josef Kraxner

Inhaltsverzeichnis:

Grundidee und Vorstellungen

Zu Beginn wurde eine Liste an möglichen Ideen erstellt. Da vorerst keine der Ideen jedem zusprach, fassten wir alle ähnlichen Features zusammen. Wir wollten die Kamera benutzen und etwas anhand des Kamera-Inputs am Bildschirm zeichnen. Das Vorhaben hatte zusammengefasst große Ähnlichkeit mit Augmented Reality Applikationen.

Wir entschieden uns, je nach verfügbarer Zeit, das Projekt in diese Richtung auszuweiten. Als Basis des Projektes brauchten wir etwas, das wir erfassen konnten. Hier kam der QR-Code in Frage, da es bereits Realisierungen davon gab.

Allgemeine Informationen zu QR Codes

Quick Response Codes, als Kurzform QR-Codes, zählt zu den zweidimensionalen Codes. Die Verwendung war hauptsächlich für Produktionslogistik gedacht, um Komponenten zu kennzeichnen und wiederzuerkennen. Entworfen wurde der QR-Code von der Firma Denso Wave.

Der Code ist als quadratische Matrix aufgebaut. In drei der vier Ecken sind sogenannte „Finder Patters“ platziert. Anhand dieser lässt sich die Ausrichtung feststellen. In größeren QR-Codes befinden sich zwischen den Daten sogenannte „Alignment Patterns“, welche für Positionskorrekturen aufgrund von hoher Datenkapazität verwendet werden.

Die Größe eines QR-Codes ist abhängig von den zu enthaltenen Daten, welche als „Version“ festgehalten wird. Version 1 stellt den kleinstmöglichen QR-Code dar, welcher die Dimension 21x21 hat. Der größtmögliche QR-Code hat die Version 40 und die Dimension 177x177. In diesem lassen sich bis zu 4296 alphanumerische Symbole speichern.

Des Weiteren ist es möglich, durch eine automatische Fehlerkorrektur Auslesefehler zu erkennen und zu korrigieren. Damit können bis zu 30% fehlerhafter Daten erkannt und korrigiert werden.

Die Verwendung von QR-Codes ist mittlerweile lizenz- und kostenfrei.

Realisierung und Implementation

Die Hauptproblematik beim Erfassen von QR-Codes liegt in der Erkennung, insbesondere von dunklen, quadratischen Flächen. Dies wird hauptsächlich durch grafische Ansätze gelöst, welche der Erkennung von Gesichtern ähnlich ist. Da diese Ansätze sehr kompliziert und allgemein sind, entschieden wir uns einen Eigenen Ansatz zu entwerfen, der auf unsere Problemstellung zugeschnitten ist.

Suchen von dunklen Bereichen im Bild:

Hierfür gibt es mehrere Möglichkeiten. Wir entschieden uns, das Bild zu rastern und jeden Rasterpunkt auf dessen Farbe abzufragen. Hierbei ist zu beachten, dass man nicht auf Schwarz und Weiß abfragt, sondern auf einen Grad der Dunkelheit, ausgedrückt durch Intensitäten von Rot, Grün und Blau.

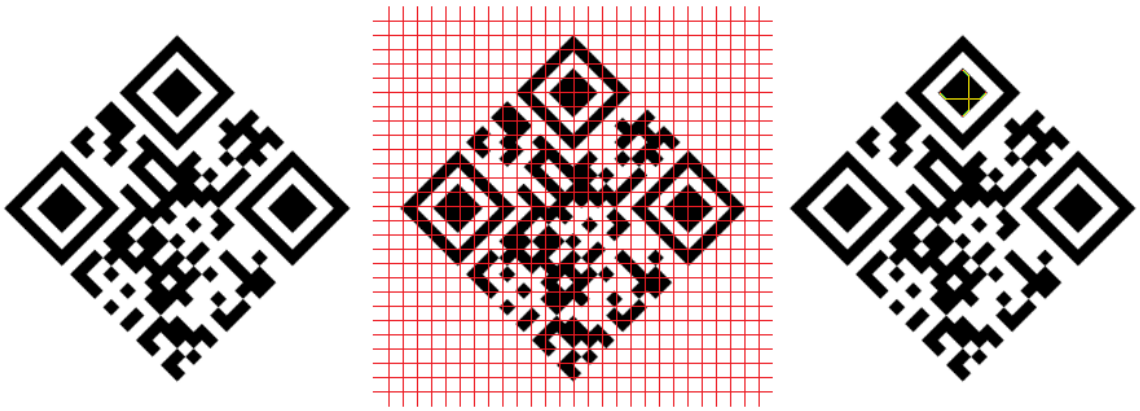
Erkennung der Form einer Fläche:

In unserem Fall sind wir an dunklen, quadratischen Flächen interessiert. Die Erkennung sollte unabhängig von der Ausrichtung und der Rotation des QR-Codes sein.

Geht man vom Idealfall aus, könnte man jede Ecke eines Quadrates sich grafisch über einen Weg in 2 Richtungen erreichen. Um schräge Kanten zu berücksichtigen wird eine dritte Bewegungsrichtung benötigt.

	Links Oben	Rechts Oben	Links Unten	Rechts Unten
Grundrichtungen	links	oben	unten	rechts
	oben	rechts	links	unten
Spezialrichtung	unten	links	rechts	oben

Jede Ecke eines Quadrates stellt eine Sackgasse dar. Werden vier erreicht, ohne die Grenzen des Bildes zu überschreiten, wissen wir, dass wir ein Viereck gefunden haben. Es wird nun überprüft, ob die Seiten des Vierecks annähernd gleich lang sind. Ist dies der Fall, wird es als Quadrat klassifiziert.



Sammeln der größten in Frage kommenden Flächen:

Es werden nun nacheinander Quadrate erkannt. Drei davon sind die Finder Patterns die wir suchen, daher werden drei Platzhalter für die Finder Patterns reserviert. Jedes neue Quadrat wird geprüft, ob es groß genug ist oder ein Duplikat darstellt. Wurde das gesamte Bild geprüft, so stellen die im Platzhalter vorhandenen Quadrate die Finder Patterns dar.

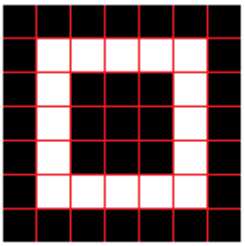
Festlegung des Koordinatensystems:

Das in der Mitte liegende Finder Pattern stellt den Ursprung des Koordinatensystems dar, die zwei anderen jeweils die X- und Y-Achse.

Anhand der Abstände unter den einzelnen Finder Patterns lässt sich herausfinden, welches der Finder Patterns der Ursprung sein muss. Anhand der Festlegung im Aufrechten Zustand werden die Achsen zugeteilt.

Normierung von Abständen:

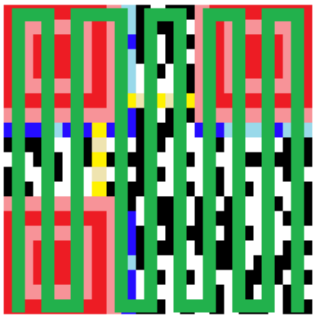
Zum Auslesen der Daten muss die Dimensionierung eines Elementes in Pixel bekannt sein. Da die Länge und Breite der Finder Patterns 7 Elemente breit ist, lässt sich dies ohne Probleme berechnen.



Da die Ausrichtung berücksichtig werden muss, werden die von der Ausrichtung abhängigen Richtungsvektoren für X- und Y-Richtung anhand der Positionen der Finder Patterns berechnet und anhand einer Elementbreite normiert. Anschließend wird noch mit einem Offset der Nullpunkt im linken oberen Eck des Ursprungs-Finder-Patterns festgelegt. Alle Schritte lassen sich mathematisch durch lineare Algebra realisieren.

Auslesevorgang und Enkodierung:

Da man aus den nun errechneten Vektoren die Position jedes Pixels bestimmen kann, ist es nun möglich, die Daten der Reihe nach auszulesen. Es wird die nächstgelegene Version berechnet, die Daten ausgelesen und für den Enkodierungsvorgang als binäre Matrix gespeichert.



Um die Daten in lesbare Symbole umzuwandeln, werden die Daten in der binären Matrix in einer bestimmten Reihenfolge ausgelesen. Davor werden Informationen über das Format und die Version gelesen.

Error Correction Level		Mask Pattern			Error Correction Bits				
1	0	1	0	1	0	0	1	0	…

Diese Informationen haben eine Länge von 15 Bit bzw. 18 Bit und enthalten das „Error Correction Level“, das „Mask Pattern“ und die „Error Correction Bits“. Da wir die Fehlerkorrektur in unserer Anwendung nicht betrachten, ist die Bitmaske das einzige was wir auslesen.

Da in der binären Matrix noch immer die strukturellen Elemente des QR-Codes vorhanden sind, müssen diese für den Auslesevorgang ausgeschlossen werden. Da die Positionen der strukturellen Elemente bekannt sind, wurden sie mit einem dritten Zustand gekennzeichnet. Die Matrix ist nun tertiär.

Mode				Character Count									Data														
0	0	1	0	0	0	0	0	1	1	1	0	1	0	1	1	0	1	0	0	0	1	0	0	0	1	0

Nach dem Auslesevorgang unter Berücksichtigung der Bitmaske liegen die Daten linear vor. Die ersten 4 Bits definieren den Auslesemodus, die darauffolgenden 9 Bits die Anzahl der enthaltenen Symbole. Danach folgen die Daten, welche mit einem der folgenden Modi ausgelesen werden:

- Numeric: nur Zahlen
- Alphanumeric: Zahlen und einfache Buchstaben
- Byte: ISO-8859-1 und UTF-8
- Kanji: Japanische Kanji Zeichen

Wurden die Daten im richtigen Modus ausgelesen, ist der Enkodierungsvorgang abgeschlossen und die Daten des QR-Codes liegen wie gewünscht vor.

Testen der Anwendung

Rasterung und Schwarz-Weiß-Interpretation

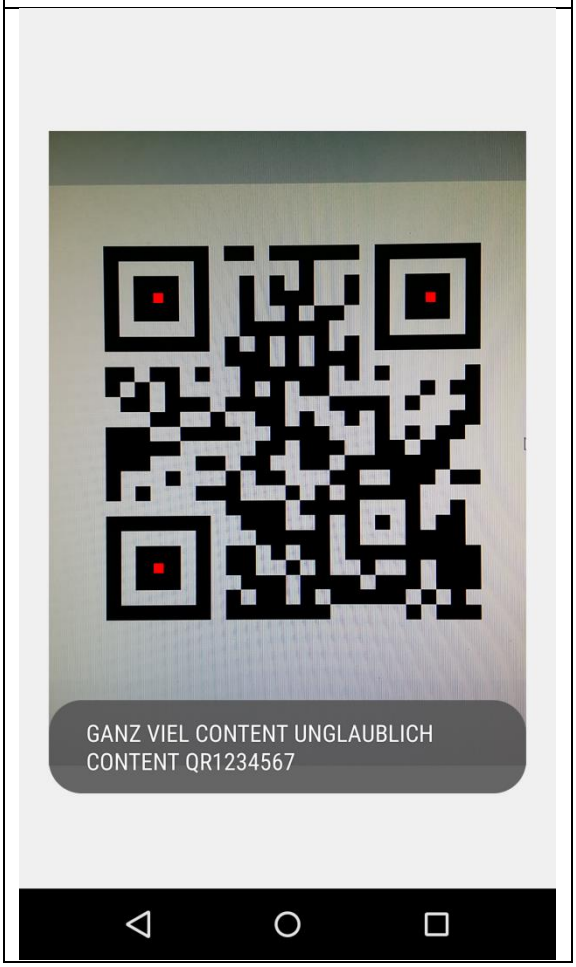
Erstellen der binären Matrix

```
Finding squares in 1362:1059...
Square 1: (89.55445270895244 * 89.55445270895244 = 8019.999999999999) @ center X: 663.5 Y: 242.5
  Edge 1: X: 603.0 Y: 226.0
  Edge 2: X: 680.0 Y: 182.0
  Edge 3: X: 647.0 Y: 303.0
  Edge 4: X: 724.0 Y: 259.0
Square 2: (89.55445270895244 * 89.55445270895244 = 8019.999999999999) @ center X: 301.5 Y: 451.5
  Edge 1: X: 241.0 Y: 435.0
  Edge 2: X: 318.0 Y: 391.0
  Edge 3: X: 285.0 Y: 512.0
  Edge 4: X: 362.0 Y: 468.0
Square 3: (89.55445270895244 * 89.55445270895244 = 8019.999999999999) @ center X: 510.5 Y: 813.5
  Edge 1: X: 450.0 Y: 797.0
  Edge 2: X: 527.0 Y: 753.0
  Edge 3: X: 494.0 Y: 874.0
  Edge 4: X: 571.0 Y: 830.0
delta_x: 29.85148423631748 delta_y: 29.85148423631748
matrix_dimension_x: 21 (21.002693898281773) matrix_dimension_y: 21 (21.002693898281773)
delta_x_vector: X: 25.852168349150293 Y: -14.925699406001135
delta_y_vector: X: 14.925699406001135 Y: 25.852168349150293
offset: X: -122.33360326545429 Y: -32.77940682944748
0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 0 0 0 0 0 0
0 1 1 1 1 1 1 0 1 0 0 1 1 0 1 0 1 1 1 1 1 0
0 1 0 0 0 0 1 0 1 1 0 1 0 0 1 0 1 0 0 0 1 0
0 1 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0 0 1 0
0 1 0 0 0 0 1 0 1 0 0 1 0 1 1 0 1 0 0 0 1 0
0 1 1 1 1 1 1 0 1 1 0 1 1 0 1 0 1 1 1 1 1 0
0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 0 0 1 0 0 1 1 1 1 1 1 1 1
0 0 1 0 0 0 0 0 1 0 0 1 1 0 0 0 1 0 0 1 0 1
0 1 0 0 0 0 0 1 0 1 1 1 1 0 0 0 0 1 0 0 0 1
1 1 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 1 1 1 1
0 1 0 0 1 0 1 1 1 0 1 0 0 1 1 1 0 0 1 1 1
0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0
1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 1
0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 0 1 1 0 0 0 0
0 1 1 1 1 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 0 0
0 1 0 0 0 1 0 1 0 0 1 0 1 1 0 1 1 1 0 0 0
0 1 0 0 0 1 0 1 0 1 0 0 0 1 1 1 0 1 0 1 1
0 1 0 0 0 1 0 1 1 0 1 1 1 1 0 1 1 1 1 0 0
0 1 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 0 0 1
0 0 0 0 0 0 0 0 1 1 0 1 0 1 1 1 1 1 1 1 0 1
No errors occured.
```

Markieren von Strukturelementen in tertiärer Matrix mit Enkodierung

```
2 2 2 2 2 2 2 2 2 2 0 0 1 0 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 1 0 0 1 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 1 0 1 1 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 1 1 1 1 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 1 0 1 0 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 1 0 0 1 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 1 0 1 1 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 1 0 0 1 2 2 2 2 2 2 2 2 2
1 0 1 1 1 1 2 1 0 0 0 0 1 1 1 1 0 1 1 1 0
0 0 1 0 1 0 2 1 0 0 0 1 0 0 1 1 0 0 0 0 0
1 0 1 1 0 1 2 0 0 1 0 1 1 0 0 0 1 1 0 0 0
1 1 0 1 1 1 2 1 1 1 1 0 1 1 1 0 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2 0 0 0 1 0 0 1 0 1 0 0 0
2 2 2 2 2 2 2 2 2 2 1 1 0 0 1 1 0 0 1 1 1 1
2 2 2 2 2 2 2 2 2 2 0 1 0 0 1 0 0 1 0 1 1 1
2 2 2 2 2 2 2 2 2 2 1 0 1 0 0 1 0 0 0 1 1 1
2 2 2 2 2 2 2 2 2 2 0 1 1 1 0 0 0 1 0 1 0 0
2 2 2 2 2 2 2 2 2 2 1 0 0 0 0 1 0 0 0 0 1 1
2 2 2 2 2 2 2 2 2 2 1 1 0 0 1 1 1 0 0 1 1 0
2 2 2 2 2 2 2 2 2 2 1 0 1 0 0 0 0 0 0 0 1 0
Mask number: 6
Mode: 2 character count: 11 data size: 195
Convert to characters ...
HELLO WORLD
```

Test auf dem Gerät



Termine und Fazit

Pairing Matrix:

Number of minutes the two members did physically co-located ping pong pair programming.			
	ThomasSchrotter	MarkusSchrotter	JosefKraxner
ThomasSchrotter		1740	1130
MarkusSchrotter	1740		0
JosefKraxner	1130	0	
SUM:	2870	1740	1130

Number of minutes the two members did remote ping pong pair programming.			
	ThomasSchrotter	MarkusSchrotter	JosefKraxner
ThomasSchrotter		690	410
MarkusSchrotter	690		0
JosefKraxner	410	0	
SUM:	1100	690	410

Overall sums in minutes:	3970	2430	1540
Sums in HH:MM:SS format	66:10:00	40:30:00	25:40:00

Timesheet:

16.06.2015	09:00	18:00	6:00:00	03:00	Implementation von ByteMode, Dokumentation, Abschluss von Iteration 4 und hochladen des Final Releases von QRTrace
14.06.2015	10:00	15:00	4:30:00	00:30	Implementation von NumericMode, Bugfixing, Trennung von Info und Content
12.06.2015	09:00	16:00	6:30:00	00:30	Abschluss von Iteration 3, Planung für Iteration 4, Besprechung der übrigen Probleme, Umsetzungsumfang angepasst
08.06.2015	10:00	12:15	2:15:00	00:00	Usability Tests, Design considerations
05.06.2015	09:30	16:15	6:15:00	00:30	Planung für Iteration 3, ausstehende Probleme, erstellen eines ersten Releases, App-Design und Usability, Umsetzungsumfang angepasst
02.06.2015	10:00	16:45	6:15:00	00:30	Anwendung der Algorithmen auf mehrere nacheinander folgenden Bildern (onPreviewFrame), weitere Tests und Anpassungen
01.06.2015	09:00	14:30	5:30:00	00:00	Kombination mit Hardware, SurfaceView und CameraPreview mit Algorithmen kombiniert, weitere Tests und Anpassungen
29.05.2015	09:00	13:30	4:30:00	00:00	Erstellen von Tests mit realen Bildern, weitere Anpassungen an den Algorithmus
26.05.2015	10:00	14:15	4:15:00	00:00	Erste Tests mit realen Bildern, Modifikation der Algorithmen zur Erfassung realer Bilder
23.05.2015	10:00	16:30	6:00:00	00:30	Planung für Iteration 2, Besprechung bisheriger Probleme, Aufwandsabschätzung, Umsetzungsumfang an derzeitigen Projektverlauf
19.05.2015	10:00	17:00	6:30:00	00:30	Funktionsfähiger Algorithmus für encodeMatrix fertig, testen des Algorithmusses, erster Versuch, convertToQRMatrix und encodeMatrix zu
18.05.2015	09:00	13:45	4:45:00	00:00	Entwurf von encodeMatrix zum Dekodieren der Informationen eines QR-Codes
15.05.2015	09:00	15:30	6:00:00	00:30	Funktionsfähiger Algorithmus für convertToQRMatrix fertig, testen des Algorithmusses
12.05.2015	10:00	15:00	4:30:00	00:30	Erste grobe Tests für Module convertToMatrix, covertToQRMatrix und encodeMatrix erstellt, Entwurf eines Algorithmusses zur Erfassung
11.05.2015	09:00	14:30	5:30:00	00:00	Startgespräch, Planung für Iteration 1, Ziele gesetzt, Projekt in Teile aufgeteilt und strukturiert, Ort und Zeit für regelmäßige Treffen
BEGINN					

Fazit:

Die Kombination von Auslesen und Erfassen von QR-Codes wurde prägend für die Applikation. Wir schafften es zwar nicht die Applikation bis zum Teil der Augmented Reality zu entwickeln, jedoch war die Applikation ein Erfolg. Es wurden fast alle Möglichkeiten von QR-Codes abgedeckt und die Erfassung funktionierte besser als erwartet.

Anmerkungen und Quellen

http://www.thonky.com/qr-code-tutorial/introduction

Zuletzt möchten wir noch anmerken, dass uns die Informationen dieser Seite sehr weitergeholfen haben, welche die Fakten rund um den QR-Code wesentlich detaillierter zusammenfasst.