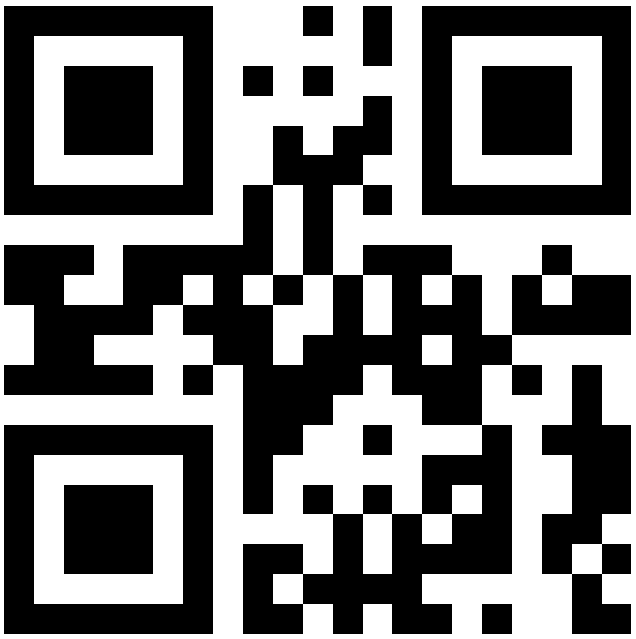

# Matlab QR-Code-Reader

## Konzept

**Modul: BZG1301 "Programmierung in Matlab/Octave"**
**Dozent: Marx Stampfli**
**Autor: Joel Holzer**
**Version: 1.0, 16.11.2015**

**Berner Fachhochschule**
Technik und Informatik
Informatik

# 1 Konzept

QR-Codes sind in der heutigen Zeit ein weit verbreitetes Medium um Menschen einen schnellen Zugriff auf Daten zu ermöglichen. Mit einem klassischen QR-Code-Scanner (Gerät) oder einem Mobiltelefon kann der QR-Code gescannt und so dessen Information ausgelesen und dem Benutzer in verständlicher Form angezeigt werden. Oft werden QR-Codes verwendet, um URLs von Webseiten zu Speichern. Beim Scan des QR-Codes mit dem Mobiltelefon wird dem Benutzer dann die entsprechende Webseite im Browser angezeigt. Weitere Anwendungsbereiche sind bei IT-Lösungen im Mobile Computing oder Internet of Things (kurz IOT) zu finden. Dort werden QR-Codes hauptsächlich verwendet um Daten zwischen verschiedenen mobilen Geräten oder Things (bei IOT) auszutauschen. QR-Codes sind aber nicht nur auf die Speicherung von Text beschränkt, sondern ermöglichen die Speicherung beliebiger binärer Daten (bis max. 2956 Byte).

Für Softwareentwickler stehen eine Vielzahl von Libraries zur Verfügung, welche die einfache Integration eines QR-Code-Readers in eigene Applikationen ermöglichen. Dazu sind meistens weder Grundwissen über den Aufbau von QR-Codes, noch Kenntnisse über die Bildverarbeitung erforderlich. Die ganze Analyse des QR-Code-Bilds und die Datenextraktion aus dem Bild werden meistens durch Third Party Libraries durchgeführt.

Auch der Autor dieser Arbeit hat bereits mehrfach Erfahrungen mit dem Einsatz von QR-Code-Reader-Libraries im Android-Umfeld gesammelt. Doch wie kann ein QR-Code ohne diese Libraries gelesen werden?

Im Rahmen dieser Arbeit wird genau dieser Frage auf den Grund gegangen. Dazu wird mit Hilfe von Matlab von Grund auf einen QR-Code-Reader entwickelt. Auf die Verwendung von Matlab QR-Code-Libraries wird verzichtet.

Die zu entwickelte Matlab-Applikation soll folgende Funktionalitäten beinhalten:

- **Angabe des Bildpfads:** Pfad zur Bilddatei (PNG), welche einen 2D QR-Code beinhaltet, kann angegeben werden.
- **Erkennung des QR-Codes in einem Bild:** Die Applikation erkennt durch die Anwendung klassischer Methoden aus der Bildanalyse den QR-Code in der angehängten Bilddatei. Der QR-Code in der Bilddatei kann nur ein Teil des Gesamtbilds sein und sich irgendwo im Bild befinden.
  Einschränkung: In einer ersten Version wird die Funktionsweise so eingeschränkt, dass der QR-Code im Bild in den Farben Schwarz/Weiss sein muss. Je nach Verlauf des Projekts wird die Applikation so erweitert, dass der QR-Code irgendwelche kontrastreichen Farben beinhalten darf.
- **Extraktion des Inhalts:** Die Applikation kann den Inhalt von QR-Codes, welche Text (Buchstaben, Zahlen, Zeichen) nach ISO-8859-1 beinhalten, aus dem QR-Code extrahieren und den Text dem Benutzer auf dem Bildschirm anzeigen.
- **Unterstützung verschiedener Versionen von QR-Codes:** Es gibt verschiedene Versionen von QR-Codes. Diese unterscheiden sich in der Anzahl Module. Je mehr Module ein QR-Code beinhaltet, desto mehr Daten können gespeichert werden. Der entwickelte QR-Code-Reader soll alle QR-Code Versionen, von Version 1 mit 21 x 21 Modulen bis hin zu Version 40 mit 177 x 177 Modulen lesen können.

Neben der entwickelten Matlab-Applikation wird eine Dokumentation erstellt, in welcher die wichtigsten Fakten zum Aufbau von QR-Codes, sowie die Umsetzung des QR-Code-Readers mit Hinweisen und Darlegungen der mathematischen Hintergründe, festgehalten werden.

# 2 Vorgehen

## 2.1 Technologien / Hilfsmittel / Tools

- **Matlab R2015a:** Entwickelt wird die Applikation mit Matlab in der Version R2015a.
- **Bildverarbeitung-Funktionen:** Nutzung der integrierten Matlab-Funktionen. Falls diese nicht ausreichen, wird auf die Matlab „Image Processing Toolbox“ zurückgegriffen.
- **Generierung von QR-Codes:** QR-Codes können mit der Webseite http://www.qrcode-generator.de/ generiert werden.
- **Literatur:** Das benötigte Wissen über QR-Codes und über die Bildverarbeitung wird von Webseiten erlangt. Auf Bücher oder andere literarische Werke wird nicht zurückgegriffen.

## 2.2 Methoden und Algorithmen

Nachfolgend ein möglicher Ansatz zur Realisierung des Algorithmus zur QR-Code-Erkennung:

1. Bild in ein Graustufen-Bild und danach in ein Binary-Bild konvertieren. Dies erhöht die Erkennungsrate des QR-Codes.
2. Die 3 Begrenzungsmuster des QR-Codes finden. Die 3 Begrenzungsmuster charakterisieren sich durch ein Modul-Verhältnis von 1:1:3:1:1 von den weissen Modulen (Blöcken) zu den schwarzen Modulen (Blöcken). Die Begrenzungsmuster sind im nachfolgenden Bild Rot markiert.

Abbildung 1 QR-Code mit Begrenzungsmuster (Rot) und Ausrichtungsmuster (Blau).
(Quelle: http://www.swisseduc.ch/informatik/theoretische_informatik/qr_codes/docs/unterlagen_lernende.pdf )

3. Nachdem die Begrenzungsmuster gefunden wurden, kann mit deren Grösse die Grösse eines einzelnen Moduls (Block) errechnet werden. Damit kann die Anzahl Module im QR-Code und somit die Version des QR-Codes herausgefunden werden.
4. Je nach Version des QR-Codes sind Ausrichtungsmuster vorhanden. Diese sind im obenstehenden Bild Blau markiert und der QR-Code-Reader findet diese anhand deren Schwarz-Weiss-Verhältnisses.
5. Alle im obigen Bild nicht Blau, Hellblau, Rot, Grün, Gelb oder Pink markierten Muster beinhalten binäre Daten. Schwarz ist gleich 1, weiss gleich 0. QR-Code-Reader geht nun das Bild Modul für Modul durch und kommt so zu einem binären String und somit zu den Daten im QR-Code.
6. Daten des Bilds werden nur mit der auf den QR-Code angewendeten Maskierungsmaske demaskiert um die Ausgangsdaten zu erhalten. Die Maskierungsmaske wird aus dem Hellblauen Teil des obenstehenden Bilds ermittelt.
7. Ausgangsdaten werden mit ISO-8859-1 dekodiert und so der Inhalt des QR-Codes in Textform erhalten.

# 3 Zeitplan

| Tätigkeit | KW 49<br>2.12 | KW 50<br>9.12 | KW 51<br>16.12 | KW 52<br>23.12 | KW 53<br>30.12 | KW 01<br>6.1 | KW 02<br>13.1 | KW 03<br>20.1 |
|---|---|---|---|---|---|---|---|---|
| Einlesen in QR-Code-Theorie | | | | | | | | |
| Realisierung: Einlesen eines Bilds und Erkennung der Begrenzungsmuster | | | | | | | | |
| Realisierung: Erkennung der übrigen standardisierten Muster und Ausrichtung des QR-Codes | | | | | | | | |
| Realisierung: Extraktion der Daten | | | | | | | | |
| Realisierung: Demaskierung der Daten | | | | | | | | |
| Realisierung: Dekodierung und Ausgabe der Daten | | | | | | | | |
| Finalisierung der Dokumentation | | | | | | | | |
| Präsentation und Abgabe | | | | | | | | |

Tabelle 1 Zeitplan

# 4 Versionskontrolle

| Version | Datum | Beschreibung | Autor |
|---|---|---|---|
| 0.1 | 11.11.2015 | - Dokument erstellt.<br>- Mit der Erstellung des Kapitel „Konzept“ begonnen. | Joel Holzer |
| 0.2 | 13.11.2015 | - Kapitel „Konzept“ fertiggestellt. | Joel Holzer |
| 0.3 | 16.11.2015 | - Kapitel „Vorgehen“ erstellt.<br>- Kapitel „Zeitplan“ erstellt. | Joel Holzer |
| 1.0 | 16.11.2015 | - Dokument finalisiert und freigegeben. | Joel Holzer |