Übung zur Vorlesung

Multimedia im Netz

Doris Hausen
Ludwig-Maximilians-Universität München
Wintersemester 2009/2010

Steganographie

- Steganographie ist die verborgene Speicherung oder Übermittlung von Informationen
- Wörtlich aus dem Altgriechischen „bedeckt schreiben“

- Steganographie braut darauf, dass Dritte nicht bemerken, dass eine Nachricht übermittelt wird.
- Zum Vergleich: In der Kryptographie kann ein Dritter wissen, dass eine Nachricht übermittelt wird, aber er kann den Inhalt nicht entziffern

Daten in Bildern verstecken

- Menschliches Auge ist gegen Bildrauschen relativ unempfindlich → Foto kann verändert werden, ohne dass die Veränderung auffällig oder störend ist

- Eine Möglichkeit: Informationen in Farbkanälen verstecken

- Ändert man in jedem Farbkanal den Wert um eins, ändert sich die Farbe im gesamten Bildpunkt um 1/255 → 0,39%

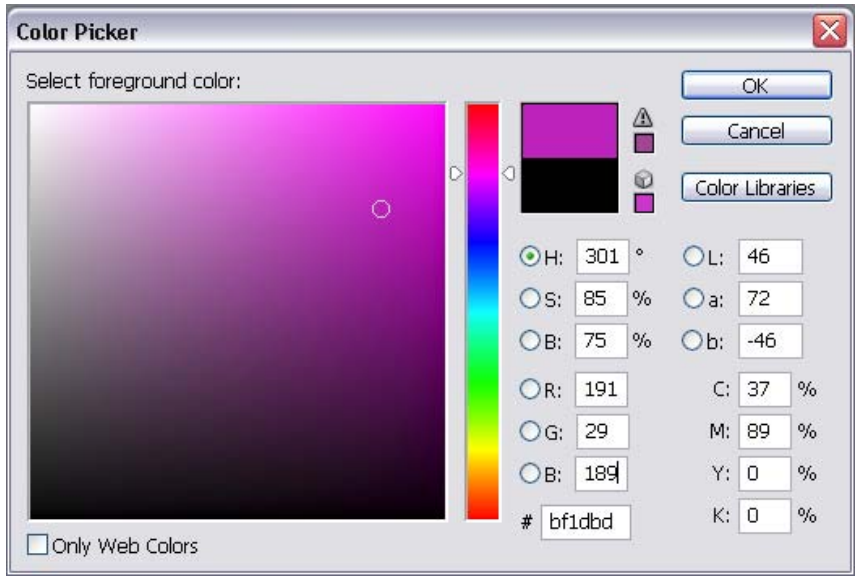
Farbkanäle manipulieren

- Ein Pixel besteht aus dem Alphakanal und drei Farbkanälen (Rot, Grün, Blau)

- Ein Kanal besteht aus 1 Byte

- Einem Buchstaben ist laut ASCII Tabelle genau eine Zahl zugeordnet, die sich logischerweise binär mit einem Byte darstellen lässt.

Beispiel



- Farbe rechts hat folgende Werte:
  - Rot: 191 → 10111111
  - Grün: 29 → 00011101
  - Blau: 189 → 10111101

  ⇨ Man kann alle Kanäle verwenden, am einfachsten ist es nur den letzten, also den Blaukanal, zu verwenden
- Der Buchstabe „A“ hat den ASCII Wert 65 → 01000001
  ⇨ Als erstes kodieren wir die erste Stelle, also „0“
- Wir kodieren das erste Bit des „A“ in die letzte Stelle des Blaukanals
  ⇨ der neue Wert des Blaukanals ist: 10111100
- Als nächstes wird das zweite Bit des „A“ also „1“ in den Blaukanal des nächsten Pixels gespeichert werden usw.

Verschiebungsoperatoren

- SHIFT RIGHT operator: >>
  - Beispiel: 01000001 >> 7 → 00000000
- SHIFT LEFT operator: <<
  - Beispiel: 01000001 << 5 → 00100000

Logische Bit-Operationen

- Und „&“
  - Beispiel:
```
  10011010
& 10101011
= 10001010
```
- Oder „|“
  - Beispiel:
```
  10011010
| 10101011
= 10111011
```
- XOR (Exklusives Oder) „^“
  - Beispiel:
```
  10011010
^ 10101011
= 00110001
```

Verstecken

Pixel: ARG10111101
A: 01000001
⇨ Ziel: ARG10111100

1. A >> 7 → 0000000
2. A & 1
   → 00000000 & 00000001 = 00000000 (↳ modifiziertes „A“)
   (Setzt die ersten sieben Stellen auf „0“; Letzte, also die von Interesse, bleibt erhalten)
3. Wert des Pixel & 1
   → ARG10111101 & 00000000001 = 00000000001 (↳ temp. Pixel)
4. Überprüfen ob der Wert des temp. Pixels jetzt gleich dem Wert des modifizierten „A“ ist
   - Wenn „Ja“ → Weiter zum nächsten Bit des „A“ und dem nächsten Pixels
   - Wenn „Nein“
     - Ist modifiziertes „A“ gleich 1 → Pixel = Pixel + 1
     - Ist modifiziertes „A“ gleich 0 → Pixel = Pixel – 1
     - Hier: mod. „A“ ist 0 → 10111101 – 1 = 10111100 (↳ Blaukanal)

Auslesen

Gesuchter Buchstabe: wird Anfangs mit 0 initialisiert
Eingelesener Pixel z.B.: ARG10111100
⇨ Ziel: erstes Bit vom ursprünglichen „A“ finden

1. Eingelesener Pixel & 1
   → ARG1011110**0** & 00000000001 = 00000000000 (↳ temp. gesuchter Buchstabe)

2. Temp. gesuchter Buchstabe << 7
   → 00000000000 << 7 = 00000000000 (↳ temp. gesuchter Buchstabe geshiftet)

3. Gesuchter Buchstabe | temp. gesuchter Buchstabe geshiftet
   → 00000000000 | 00000000000 = 00000000000 (↳1.Bit vom gesuchten Buchstaben; wird beim nächsten Durchlauf als Wert für „Gesuchter Buchstabe“ verwendet)