Perl Mongers im Ruhrgebiet

Ruhr.pm

Objekterkennung auf Bildern mit OpenCV und Image::ObjectDetect

Autor: Simon Wilper

EMail: simon AT ruhr.pm.org

Datum: 11. März 2008

http://ruhr.pm.org/

Was ist OpenCV?

- Open Computer Vision
- Aufgabenbereiche:
  - Human-Computer Interaction (HCI)
  - Objektidentifikation
  - Objektverfolgung
  - Gesichtserkennung
  - Bewegungserkennung

Ruhr.pm

Das Prinzip

- Haar-Maß (Alfred Haar 1885-1933)
- Kontraste zwischen Regionen werden kodiert
- Training mittels einigen Hundert positiven und negativen Beispielen
- Ergebnis: Cascade-Datei

Ruhr.pm

Image::ObjectDetect

- Konstruktor new($cascade)
  - Erstellt eine neue Instanz des Image::Detect Objects
- Methode detect($file)
  - Detektiert Objekte entsprechend der Cascade-Datei in $cascade.
  - Rückgabe einer Liste aller erkannten Objekte
  - Elemente sind Hashrefs mit Keys:
    - x, y, width, height: Dimension des erkannten Objekts

Ruhr.pm

Beispiele

- #1 Anzahl erkannter Objekte
- #2 Gruppenbilder aussortieren
- #3 Position erkannter Objekte ausgeben
- #4 Erkannte Objekte markieren
- #5 Face-Cropping

Beispiel #1: Anzahl erkannter Objekte

```
#!/usr/bin/perl

use warnings;
use strict;

use Image::ObjectDetect;

if ( @ARGV < 1 ) {
        die( 'Kein Bild angegeben' );
}

# Cascade-Datei fuer Gesichter, Frontalaufnahme
my $cascade = '/usr/share/...xml';

# Instanz erzeugen
my $detector = new Image::ObjectDetect( $cascade );

# Erkannte Objekte (hier Gesichter) zurueckgeben
my @faces = $detector->detect( $ARGV[0] );

# Anzahl erkannter Objekte ausgeben
print 'Erkannte Objekte: ' .
        scalar( @faces ) . "\n";
```

scalar(@liste) erzwingt skalaren Kontext, um hier die Anzahl der Elemente in einer Liste auszugeben.

Beispiel #2: Gruppenbilder aussortieren

```
# Ab wie vielen Personen ist es eine Gruppe?
my $threshold = 3;

# Verzeichnis erzeugen fuer Gruppenbilder
mkdir "gruppenbilder" if ( ! -d "gruppenbilder" );

# Instanz erzeugen
my $detector = new Image::ObjectDetect( $cascade );

# Verzeichnis einlesen via IO::Dir
my $dir = IO::Dir->new( '.' );

while ( defined( $_ = $dir->read() ) ) {
        next if( /^\./ || -d $_ );
        my @faces = $detector->detect( $_ );

        # Bild kopieren bei Anzahl Gesichter
        # groeßergleich Schwelle
        if ( scalar(@faces) >= $threshold ) {
                copy( $_, "gruppenbilder/$_" );
                print "Bild: $_ kopiert\n";
        }
}
```

IO::Dir: Sequentieller Zugriff auf Verzeichnisse

File::Copy: Dateien kopieren

Ruhr.pm

Beispiel #3: Position von erkannten Objekten ausgeben

```
my $i=0;
while ( defined( $_ = $dir->read() ) ) {
        next if( /^\./ || -d $_ );
        my @faces = $detector->detect( $_ );

        print "Bild: $_\n";
        foreach my $face (@faces) {
                my( $x, $y, $w, $h ) = (
                $face->{x},
                $face->{y},
                $face->{width},
                $face->{height}
                );

        printf(
        "\t#%02d: Pos: % 4d, % 4d / Dim: % 4d by % 4d\n",
        ++$i, $x, $y, $w, $h
        );
        }
}
```

my(...) deklariert hier mehrere Variablen in einem Aufruf und fuellt diese mit den in entsprechender Reihenfolge mit den Listenargumenten auf der RHS.

Beispiel #4: Erkannte Objekte markieren mit Image::Magick

```
while ( defined( $_ = $dir->read() ) ) {
        next if( /^\./ || -d $_ );

        # ImageMagick-Instanz
        my $IM = Image::Magick->new;

        my $targetImage = "$markedDir/$_";
        my @faces = $detector->detect( $_ );

        print "Bild: $_\n";
        $IM->Read( $_ );
        foreach my $face (@faces) {
                my( $x, $y, $w, $h ) = (
                        $face->{x}, $face->{y},
                        $face->{width}, $face->{height});

        $IM->Draw(
        primitive => 'rectangle',
        points          => "$x,$y ".($x+$w).','.($y+$h),
        stroke          => 'red',
        strokewidth     => 2,
        antialias => 1,
        fillcolor => 'rgba(100%,100%,0%,0.3)'
        );
        }
        $IM->Write( $targetImage );
        print "\tObjekte markiert: " . scalar(@faces) . "\n";
        undef $IM;
}
```

$IM->Draw(...)
zeichnet primitive Zeichenobjekte anhand der Parameter, die als Argumente uebergeben werden.

Ruhr.pm

Beispiel #5: Cropping

```
while ( defined( $_ = $dir->read() ) ) {
        next if( /^\./ || -d $_ );
        my @faces = $detector->detect( $_ );

        print "Bild: $_\n";
        foreach my $face (@faces) {
                my $IM = Image::Magick->new;
                $IM->Read( $_ );
                my( $x, $y, $w, $h ) = (
                        $face->{x},
                        $face->{y},
                        $face->{width},
                        $face->{height}
                );

                my $croppedFace = $IM->Crop(
                        geometry => $w.'x'.$h.'+'.$x.'+'.$y
                );

                $IM->Write( "$outDir/".(++$i)."-".$_ );
        }
        print "\tObjekte extrahiert: " . scalar(@faces) . "\n";
}
```

$IM->Crop(...)
schneidet das geladene Bild zu.

Vielen Dank fuer Eure Aufmerksamkeit