

# Ein Programm zur Erstellung von Klassifikatoren zur Objekterkennung in Bildern

Michael Stahr

12. August 2013

# Inhaltsverzeichnis

Inhaltsverzeichnis	I
Abkürzungsverzeichnis	II
Abbildungsverzeichnis	II
Tabellenverzeichnis	III
<b>1 Einleitung</b>	<b>1</b>
<b>2 Vorstellung des Konsolenprogramms XML Builder</b>	<b>2</b>
2.1 Programmstart mit Übergabe von Konsolenparametern . . . . .	2
2.2 Benutzung des Teilprogramms <i>Objectmaker</i> . . . . .	4
2.3 Benutzung des Teilprogramms <i>Createsamples</i> . . . . .	5
2.4 Benutzung des Teilprogramms <i>Traincascade</i> . . . . .	6
<b>3 Vorstellung des GUI-Programms XML Builder</b>	<b>8</b>
3.1 <i>Programmstart</i> . . . . .	8
3.2 Benutzung des Teilprogramms <i>Objectmaker</i> . . . . .	8
3.3 Benutzung des Teilprogramms <i>Createsamples</i> . . . . .	9
3.4 Benutzung des Teilprogramms <i>Traincascade</i> . . . . .	10

## Abbildungsverzeichnis

1	Konsolenstart mit Parameterübergabe . . . . .	4
2	Start des Teilprogramms <i>Objectmaker</i> . . . . .	4
3	Beispiel für <i>Objectmaker</i> . . . . .	5
4	Start des Teilprogramms <i>Createsamples</i> . . . . .	6
5	Start des Teilprogramms <i>Traincascade</i> . . . . .	7
6	Start GUI-Programm . . . . .	8
7	Beispiel für GUI-Objectmaker . . . . .	9
8	Beispiel für GUI-Createsamples . . . . .	10
9	Beispiel für GUI-Traincascade . . . . .	11

# Tabellenverzeichnis

1	Kommandozeilenparameter bei Programmstart . . . . .	3
---	---	---

# 1 Einleitung

In der digitalen Bildverarbeitung stellt sich häufig die Aufgabe bestimmte Objekte in Bildern zu finden. Dafür bietet sich das Verfahren der Mustererkennung an. Das diese Aufgaben in der Praxis bereits gelöst werden können, zeigen z.B. die Gesichtserkennung oder die Erkennung von Fahrzeugen aus der Frontansicht wie in [IIS13] oder der Heckansicht in [Deg12].

Hierfür werden Klassifikatoren eingesetzt, die anhand eines umfangreichen Datenmaterials aus positiven und negativen Testbildern (jeweils >1000 Stück) durch maschinelle Lernverfahren trainiert und somit erzeugt werden können. Als Positive versteht man hierbei diejenigen Testbilder, in denen das zu erkennende Objekt vorkommt und als Negative, die Bilder in denen alles - außer das interessierende Objekt - auftauchen darf. Genauere Informationen zum maschinellen Lernen und Trainieren eines Klassifikators kann man z.B. in [VJ13] erfahren. Das Sortieren und Überprüfen der Testbilder sowie die Merkmalsextraktion aus den Positiven muss hauptsächlich manuell erfolgen und erfordert daher eine Menge Zeit und Konzentration. Erst danach kann der Algorithmus zum Einsatz kommen, der den Klassifikator erstellt. Der Klassifikator wird in Form einer XML-Datei bereit gestellt, wodurch das hier vorgestellte Programm auch seinen Namen „XML Builder“ trägt.

In der freien Programmbibliothek OpenCV stehen die hierfür benötigten Algorithmen zur Verfügung. Diese sind im wesentlichen drei Stück und wurden mit der Programmiersprache C++ implementiert. Als erstes verwendet man das Programm *Objectmaker* aus [Ach13] um aus den positiven Testbildern das interessierende Objekt zu extrahieren. Als zweites wird mit dem Programm *Createsamples* aus den extrahierten Bildausschnitten eine VEC-Datei erstellt, die für das dritte Programm *Traincascade* benötigt wird um sodann den Klassifikator zu trainieren. Nähere Information diesbezüglich findet man in [OPE13] und [Deg12].

Die erwähnten drei Programme stehen in Form einer Konsolenanwendung zur Verfügung. Aus diesem Grunde ist das Programm in Abschnitt 2 ebenfalls ein Konsolenprogramm. Momentan wird eine äquivalente Anwendung mit grafischen Nutzerschnittstellen (GUI) entwickelt, welche eine komfortablere Bedienbarkeit für den Nutzer bieten wird. In Abschnitt 3 erfolgt eine Vorstellung dieses Programms.

## 2 Vorstellung des Konsolenprogramms XML Builder

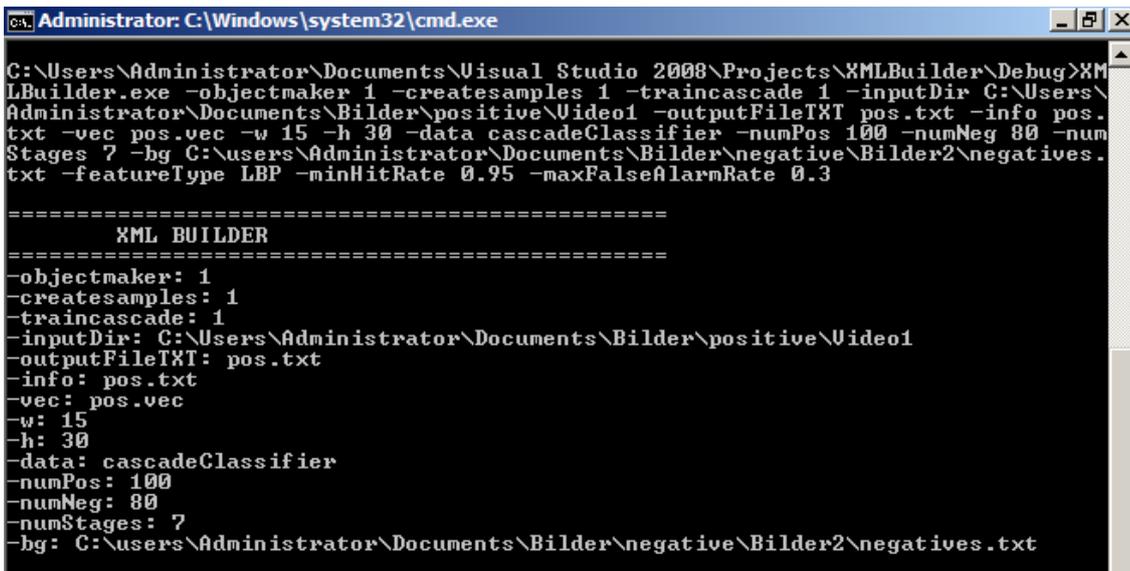
### 2.1 Programmstart mit Übergabe von Konsolenparametern

Mit XML Builder liegt ein Konsolenprogramm für Windows vor, das die zuvor beschriebenen drei Programmteile vereint. Das Programm wird über die Konsole mit dem Aufruf `XMLBuilder.exe` gestartet. Beim Aufruf des Programms können bzw. müssen u.U. bestimmte Parameter angegeben werden. Die wesentlichen Parameter sind in Tab. 1 aufgelistet, deren Bedeutung ebenfalls erläutert wird. Mit `default` wird der initialisierte Wert des Parameters angegeben. Es können allerdings noch zusätzliche Parameter gesetzt werden, wie z.B. die Verwendung von Hintergrundbildern, Rotationswinkel der Objekte und noch mehr. Informationen zu deren Verwendung können aus [OPE13] unter dem Punkt *Positive Samples* entnommen werden.

Parameter	Wert	Bedeutung
-objectmaker	default: 0; 1	0: Objectmaker wird nicht verwendet, 1: Objectmaker wird verwendet
-createsamples	default: 0; 1	0: Createsamples wird nicht verwendet, 1: Createsamples wird verwendet
-traincascade	default: 0; 1	0: Traincascade wird nicht verwendet, 1: Traincascade wird verwendet
-inputDir	!	Verzeichnispfad in dem die positiven Testbilder liegen, <b>muss</b> angegeben werden, wenn das Teilprogramm Objectmaker verwendet wird
-outputFileTXT	default: untitled.txt	Bezeichnung der Ausgabedatei des Objectmakers, steht zur Wiederverwendung für Createsamples zur Verfügung
-info	default: untitled.txt	Bezeichnung der zu verwendenden TXT-Datei die von Objectmaker erstellt wurde; diese Datei <b>muss</b> vorliegen, wenn das Teilprogramm Createsamples verwendet wird
-vec	default: untitled.vec	Bezeichnung der VEC-Datei, die von Createsamples erzeugt wird; diese Datei <b>muss</b> vorliegen, wenn das Teilprogramm Traincascade verwendet wird
-w	default: 24	Breite bzw. Höhe, auf die der extrahierte Bildteil
-h	default: 24	für die VEC-Datei skaliert wird
-data	default: classifier	Bezeichnung des Verzeichnisses in dem u.a. die XML-Datei des Klassifikators liegt
-numPos	default: 1000	Anzahl der Positiven für das Training
-numNeg	default: 1000	Anzahl der Negativen für das Training
-numStages	default: 12	Anzahl der Stufen des Klassifizierers
-bg	!	Bezeichnung der TXT-Datei für die Negativen; <b>muss</b> angegeben werden für Traincascade
-featureType	default: HAAR; LBP	Auswahl des Feature Typs, Haarlike Feature oder Local Binary Pattern (LBP ist schneller!)
-minHitRate	default: 0.995	Rate für die richtige Positiverkennung unter den Positiven pro Klassifikatorstufe (für alle Stufen gilt: $\text{minHitRate}^{\text{numStages}}$ )
-maxFalseAlarmRate	default: 0.5	Rate für die falsche Positiverkennung unter den Negativen pro Klassifikatorstufe (für alle Stufen gilt: $\text{maxFalseAlarmRate}^{\text{numStages}}$ )

Tabelle 1: Kommandozeilenparameter bei Programmstart

Ein Programmstart von XML Builder könnte wie in Abb. 1 aussehen:



```

Administrator: C:\Windows\system32\cmd.exe
C:\Users\Administrator\Documents\Visual Studio 2008\Projects\XMLBuilder\Debug>XMLBuilder.exe -objectmaker 1 -createsamples 1 -traincascade 1 -inputDir C:\Users\Administrator\Documents\Bilder\positive\Video1 -outputFileTXT pos.txt -info pos.txt -vec pos.vec -w 15 -h 30 -data cascadeClassifier -numPos 100 -numNeg 80 -numStages 7 -bg C:\users\Administrator\Documents\Bilder\negative\Bilder2\negatives.txt -featureType LBP -minHitRate 0.95 -maxFalseAlarmRate 0.3

=====
XML BUILDER
=====
-objectmaker: 1
-createsamples: 1
-traincascade: 1
-inputDir: C:\Users\Administrator\Documents\Bilder\positive\Video1
-outputFileTXT: pos.txt
-info: pos.txt
-vec: pos.vec
-w: 15
-h: 30
-data: cascadeClassifier
-numPos: 100
-numNeg: 80
-numStages: 7
-bg: C:\users\Administrator\Documents\Bilder\negative\Bilder2\negatives.txt

```

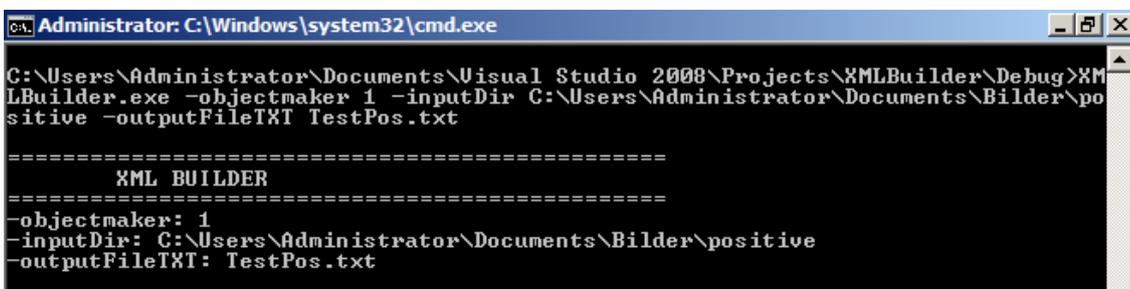
Abbildung 1: Konsolenstart mit Parameterübergabe

## 2.2 Benutzung des Teilprogramms *Objectmaker*

Es besteht die Möglichkeit nur das Teilprogramm *Objectmaker* auszuführen. Dazu sollte der Konsolenparameter `-objectmaker` auf 1 gesetzt werden, da zunächst alle drei Teilprogramm ausgeschaltet (0) sind. Die Angabe des Konsolenparameters `-inputDir` ist dann zwingend erforderlich, da aus diesem Verzeichnis die positiven Testbilder zur Anzeige und Merkmalsextraktion verwendet werden. Die Angabe des Parameters `-outputFileTXT` ist optional. Sollte hier keine explizite Angabe erfolgen, so wird die vom Teilprogramm *Objectmaker* erzeugte Ausgabedatei mit `untitled.txt` bezeichnet. Eine sinnvolle Bezeichnung der Ausgabedatei ist empfehlenswert, da diese für das Teilprogramm *Createsamples* anschließend immer wieder verwendet werden kann.

**Achtung:** Eine bereits vorhandenen Ausgabedatei (\*.txt) wird durch die Ausführung von *Objectmaker* ersetzt. Um dies zu verhindern, sollte über den Parameter `-outputFileTXT` eine andere Bezeichnung angegeben werden.

Ein Programmstart könnte wie in Abb. 2 aussehen:



```

Administrator: C:\Windows\system32\cmd.exe
C:\Users\Administrator\Documents\Visual Studio 2008\Projects\XMLBuilder\Debug>XMLBuilder.exe -objectmaker 1 -inputDir C:\Users\Administrator\Documents\Bilder\positive -outputFileTXT TestPos.txt

=====
XML BUILDER
=====
-objectmaker: 1
-inputDir: C:\Users\Administrator\Documents\Bilder\positive
-outputFileTXT: TestPos.txt

```

Abbildung 2: Start des Teilprogramms *Objectmaker*

Anschließend werden die Bilder angezeigt, die sich im Verzeichnis befinden, welches mit dem Parameter `-inputDir` angegeben wurde. Der Nutzer kann nun aus den Bildern das gewünschte Merkmal

extrahieren. Dazu wird mit der „Maus“ ein Rechteck um den entsprechenden Bildausschnitt gezogen. Danach muss mit der „Leertaste“ eine Bestätigung des Bildausschnittes erfolgen. Mit der Taste „b“ wird der bestätigte Bildausschnitt in der Ausgabedatei festgehalten und das nächste Bild angezeigt. Über die „ESC“-Taste kann die Ausführung des Programms abgebrochen werden. Informationen zum Aufbau der Ausgabedatei können in [Deg12] nachgelesen werden.

In Abb. 3 wird beispielsweise eine Lichtsignalanlage im Zustand „grün“ extrahiert. Symbolisiert wird der gewählte Bildausschnitt durch ein magenta farbenes Rechteck. In der Kopfzeile des Fensters befinden sich kurze Informationen zur Programmsteuerung bzgl. der oben erwähnten Tasten.



Abbildung 3: Beispiel für *Objectmaker*

## 2.3 Benutzung des Teilprogramms *Createsamples*

Um nur das Teilprogramm *Createsamples* zu nutzen, muss der Konsolenparameter `-createsamples` auf 1 gesetzt werden. Für eine Programmausführung muss dann aber eine mit `untitled.txt` bezeichnete oder die mit dem Parameter `-info` angegebene Datei (Ausgabedatei von *Objectmaker*) vorhanden sein. Die Angabe einer Bezeichnung für die von *Createsamples* erzeugte Ausgabedatei ist ebenfalls wieder optional. Sollte keine Angabe erfolgen, so wird die Datei `untitled.vec` genannt. Auch hier ist eine sinnvolle Bezeichnung zu empfehlen, da diese für das Teilprogramm *Traincascade* immer wieder genutzt werden kann.

**Achtung:** Eine bereits vorhandenen Ausgabedatei (`*.vec`) wird durch die Ausführung von *Createsamples* ersetzt. Um dies zu verhindern, sollte über den Parameter `-vec` eine andere Bezeichnung angegeben werden.

Die Angabe der Parameter `-w` und `-h` ist optional. Sollte keine Angabe erfolgen, wird jeweils der Wert 24 verwendet. Diese Werte legen fest, auf welche einheitliche Bildgröße die extrahierten Bildausschnitte skaliert werden, die sich in der `*.txt`-Ausgabedatei von *Objectmaker* befinden.

Eine Ausführung des Teilprogramms *Createsamples* könnte wie in Abb. 4 aussehen:

```
C:\Users\Administrator\Documents\Visual Studio 2008\Projects\XMLBuilder\Debug>XML
Builder.exe -createsamples 1 -info positiveVorfahrt.txt -vec posVorf.vec -w 20
-h 20

=====
XML BUILDER
=====
-createsamples: 1
-info: positiveVorfahrt.txt
-vec: posVorf.vec
-w: 20
-h: 20

=====
CREATESAMPLES
=====
Info file name: positiveVorfahrt.txt
Img file name: <NULL>
Vec file name: posVorf.vec
BG file name: <NULL>
Num: 1000
BG color: 0
BG threshold: 80
Invert: FALSE
Max intensity deviation: 40
Max x angle: 1.1
Max y angle: 1.1
Max z angle: 0.5
Show samples: FALSE
Width: 20
Height: 20
Create training samples from images collection...
positiveVorfahrt.txt(31) : parse errorDone. Created 30 samples

C:\Users\Administrator\Documents\Visual Studio 2008\Projects\XMLBuilder\Debug>
```

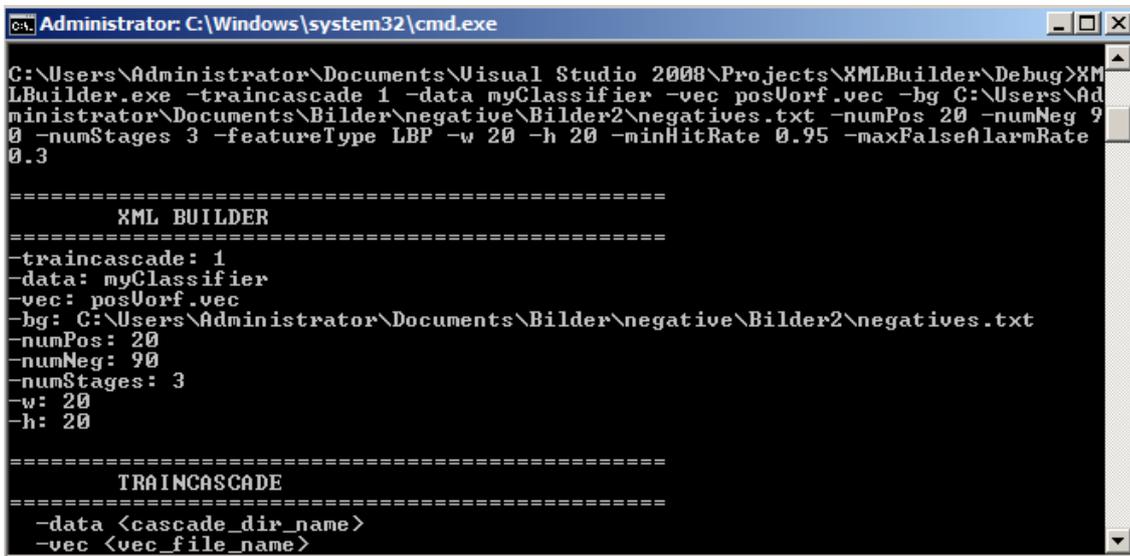
Abbildung 4: Start des Teilprogramms *Createsamples*

## 2.4 Benutzung des Teilprogramms *Traincascade*

Wenn man nur das Teilprogramm *Traincascade* nutzen möchte, sollte der Konsolenparameter `-traincascade` auf 1 gesetzt werden. Für eine Programmausführung muss eine mit `untitled.vec` bezeichnete oder die mit dem Parameter `-vec` angegebene Datei (Ausgabedatei von *Createsamples*) vorhanden sein. Zudem muss die `*.txt` Datei für die negativen Bilder existieren, die mit dem Parameter `-bg` anzugeben ist. Die Angabe einer Bezeichnung für das Verzeichnis, indem die Datei mit dem Namen `cascade.xml` abgelegt wird, ist optional. Sollte keine Angabe erfolgen, so wird der Ordner `classifier` genannt. In diesem Verzeichnis befinden sich zudem noch andere XML-Dateien. In diesen befinden sich Angaben zu den Parametern, die für das Training des Klassifizierers gewählt wurden und die Zwischenstufen des Trainings. Genauere Informationen können in [OPE13] eingesehen werden.

**Achtung:** Eine bereits vorhandener Klassifikator (*cascade.xml*) wird durch die Ausführung von *Traincascade* ersetzt. Um dies zu verhindern, sollte über den Parameter *-data* eine eigene Bezeichnung angegeben werden, sodass ein neues Verzeichnis für den Klassifikator angelegt wird.

Eine Ausführung des Teilprogramms *Traincascade* könnte wie in Abb. 5 aussehen:



```
C:\Users\Administrator\Documents\Visual Studio 2008\Projects\XMLBuilder\Debug>XMLBuilder.exe -traincascade 1 -data myClassifier -vec posVorf.vec -bg C:\Users\Administrator\Documents\Bilder\negative\Bilder2\negatives.txt -numPos 20 -numNeg 90 -numStages 3 -featureType LBP -w 20 -h 20 -minHitRate 0.95 -maxFalseAlarmRate 0.3

=====
XML BUILDER
=====
-traincascade: 1
-data: myClassifier
-vec: posVorf.vec
-bg: C:\Users\Administrator\Documents\Bilder\negative\Bilder2\negatives.txt
-numPos: 20
-numNeg: 90
-numStages: 3
-w: 20
-h: 20

=====
TRAINCASCADE
=====
-data <cascade_dir_name>
-vec <vec_file_name>
```

Abbildung 5: Start des Teilprogramms *Traincascade*

## 3 Vorstellung des GUI-Programms XML Builder

### 3.1 Programmstart

Das Programm wird durch Aufruf der Datei `XML_Builder.exe` gestartet. Anschließend hat der Nutzer drei Möglichkeiten. Er kann ein neues Projekt anlegen, ein bereits vorhandenes Projekt öffnen oder das Programm wieder beenden. Es erscheint dazu das folgende Fenster.

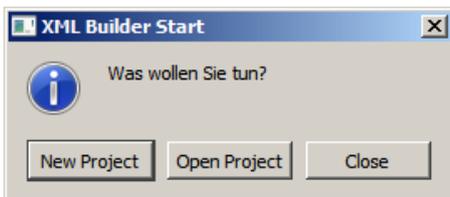


Abbildung 6: Start GUI-Programm

Möchte man ein neues Projekt anlegen, so muss der Nutzer zunächst ein Verzeichnis auswählen, indem das neue Projekt angelegt wird. Anschließend sollte ein passender Projektname vergeben werden. Wird kein Projektname vergeben, so wird das Projekt mit `Untitled` bezeichnet. Sollte der Projektname bereits vergeben sein, so wird am Ende der Bezeichnung eine Zahl (1,2,...) angehängen, je nach dem wie oft diese Projektbezeichnung bereits existiert. Die höchste Zahl steht dann für das aktuell angelegte Projekt. Anschließend wird in einer Messagebox der Verzeichnispfad zum angelegten Projekt nochmal ausgegeben.

Möchte man ein vorhandenes Projekt öffnen, so muss der Nutzer den entsprechenden Projektordner auswählen. Danach wird ebenfalls in einer Messagebox der Verzeichnispfad zum gewählten Projekt nochmal ausgegeben.

### 3.2 Benutzung des Teilprogramms *Objectmaker*

Beim Teilprogramm *Objectmaker* muss ein **Verzeichnispfad** angegeben werden, indem sich die positiven Testbilder befinden. Voreingestellt ist hier zunächst das Projektverzeichnis mit dem Unterverzeichnis `./Testbilder`. In diesem befinden sich die Bilder, die durch das Teilprogramm *Testbilder* erzeugt werden. Das Teilprogramm *Testbilder* wurde allerdings noch nicht implementiert, so dass das Unterverzeichnis zunächst leer ist. Man kann als Testbilder aber auch ein beliebiges anderes Verzeichnis wählen, indem sich Bilder mit dem interessierenden Merkmal befinden.

Weiterhin sollte eine angemessene Bezeichnung der **Ausgabedatei** erfolgen, da diese im Weiteren wiederverwendet wird. Voreingestellt ist hier die Bezeichnung `untitled.txt`. Die Ausgabedatei des Teilprogramms *Objectmaker* wird für das nächste Teilprogramm *Createsamples* als Eingabedatei benötigt.

Durch Betätigung des Start-Buttons wird das erste Bild aus dem Verzeichnispfad der Testbilder angezeigt. Mit der Maus kann das interessierende Merkmal aus dem Testbild extrahiert werden. Dazu

wird mit der linken Maustaste der linke obere Eckpunkt und danach mit der rechten Maustaste der untere rechte Eckpunkt eines Rechtecks festgelegt. Mit dem Button **Bestätigen** wird der gewählte Bildbereich in die Ausgabedatei geschrieben, so wie es in [Deg12] erläutert wird. Mit dem **Weiter**-Button wird das nächste Bild aus dem Verzeichnispfad der Testbilder dargestellt. Mit dem **Stop**-Button kann das Teilprogramm beendet werden. In Abbildung 7 wird ein Beispiel zur Extrahierung einer Lichtsignalanlage im Zustand „grün“ gezeigt.

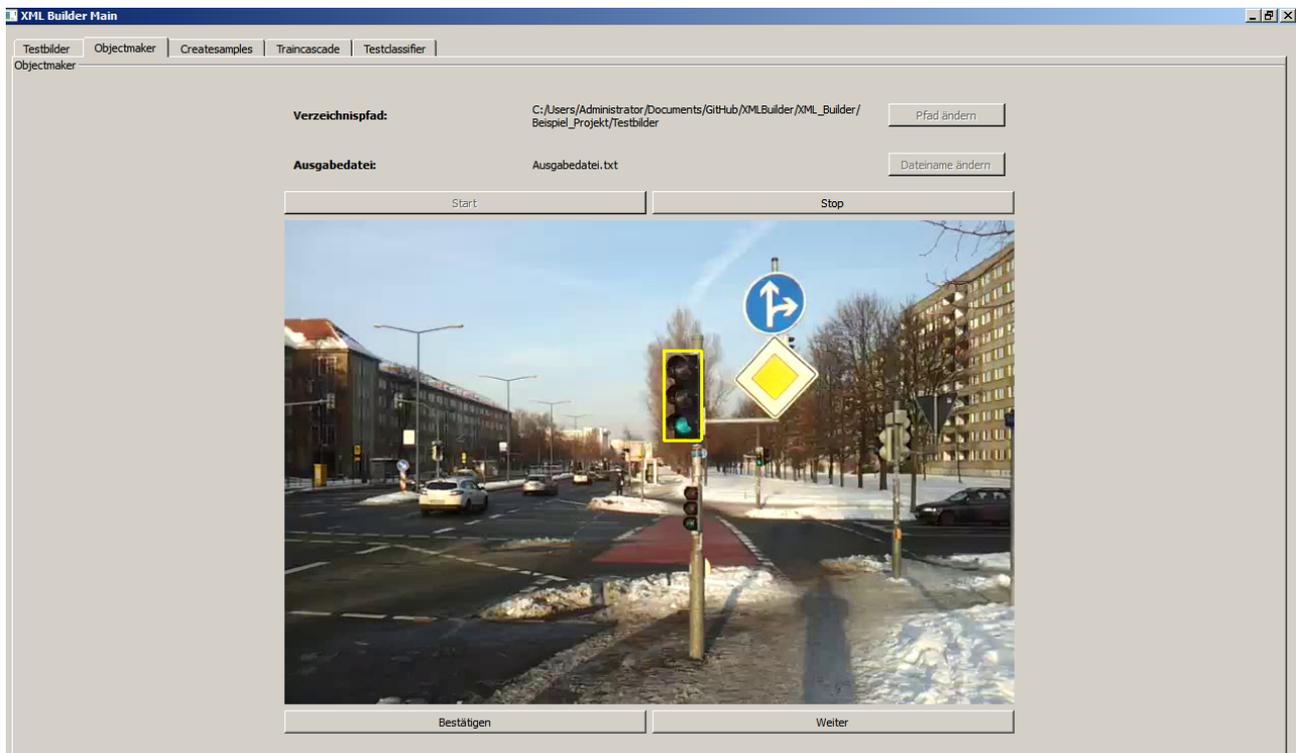


Abbildung 7: Beispiel für GUI-Objectmaker

### 3.3 Benutzung des Teilprogramms *Createsamples*

Der Nutzer hat beim Teilprogramm *Createsamples* vier Parameter zu wählen. Die **Eingabedatei** ist eine \*.txt Datei, die durch das vorhergehende Teilprogramm *Objectmaker* erzeugt wurde. In dieser befinden sich die interessierenden Bildausschnitte aus den positiven Testbildern, aus denen eine \*.vec Datei erzeugt wird, deren Name mit dem Parameter **Ausgabedatei** bestimmt wird. Dieser sollte sinnvoll vergeben werden, da diese Datei im nachfolgenden Teilprogramm *Traincascade* wieder verwendet wird. Die **Breite und Höhe der Skalierung** des ausgewählten Bildausschnittes kann im Bereich von 10 bis 50 Pixel angegeben werden. Voreingestellt sind hier jeweils 24 Pixel.

Durch Betätigung des **Start**-Buttons wird durch die Programmbibliothek OpenCV eine \*.vec Datei erzeugt.

>> Hier gibt es allerdings Probleme beim Einbinden der Funktionen aus der externen Bibliothek in das Qt-Projekt. <<

In Abbildung 8 wird das Teilprogramm *Createsamples* dargestellt.

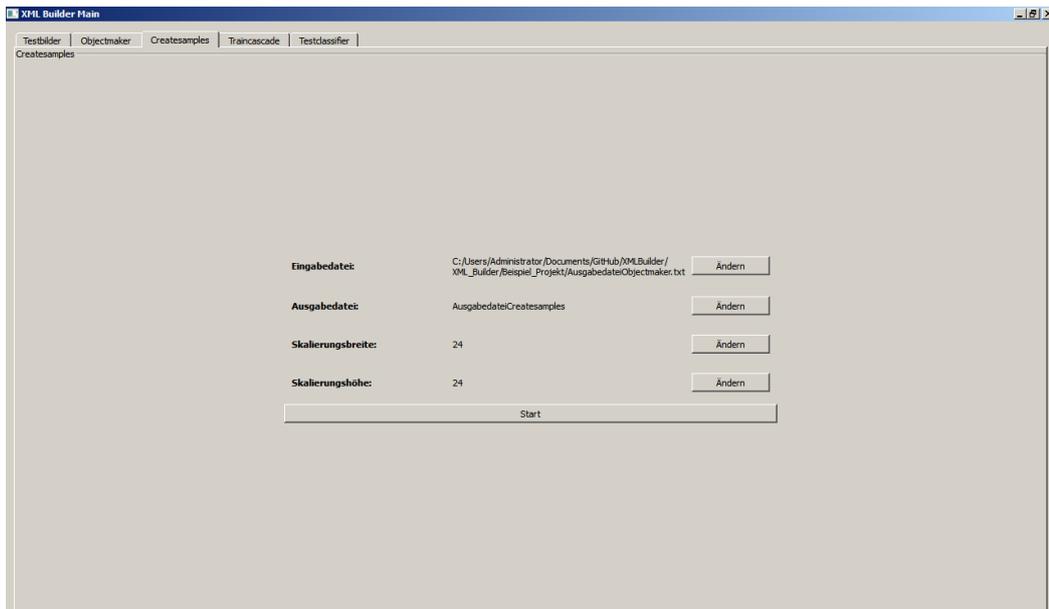


Abbildung 8: Beispiel für GUI-Createsamples

### 3.4 Benutzung des Teilprogramms *Traincascade*

Bei der Benutzung des Teilprogramms *Traincascade* sind neun Parameter zu wählen. Die Bedeutung der einzelnen Parameter kann aus Tabelle 1 entnommen werden. Durch Betätigung des Start-Buttons wird durch die Programmbibliothek OpenCV das Training zur Erzeugung einer \*.xml Datei ausgeführt.

>> Hier gibt es allerdings Probleme beim Einbinden der Funktionen aus der externen Bibliothek in das Qt-Projekt. <<

In Abbildung ist das Teilprogramm *Traincascade* dargestellt.

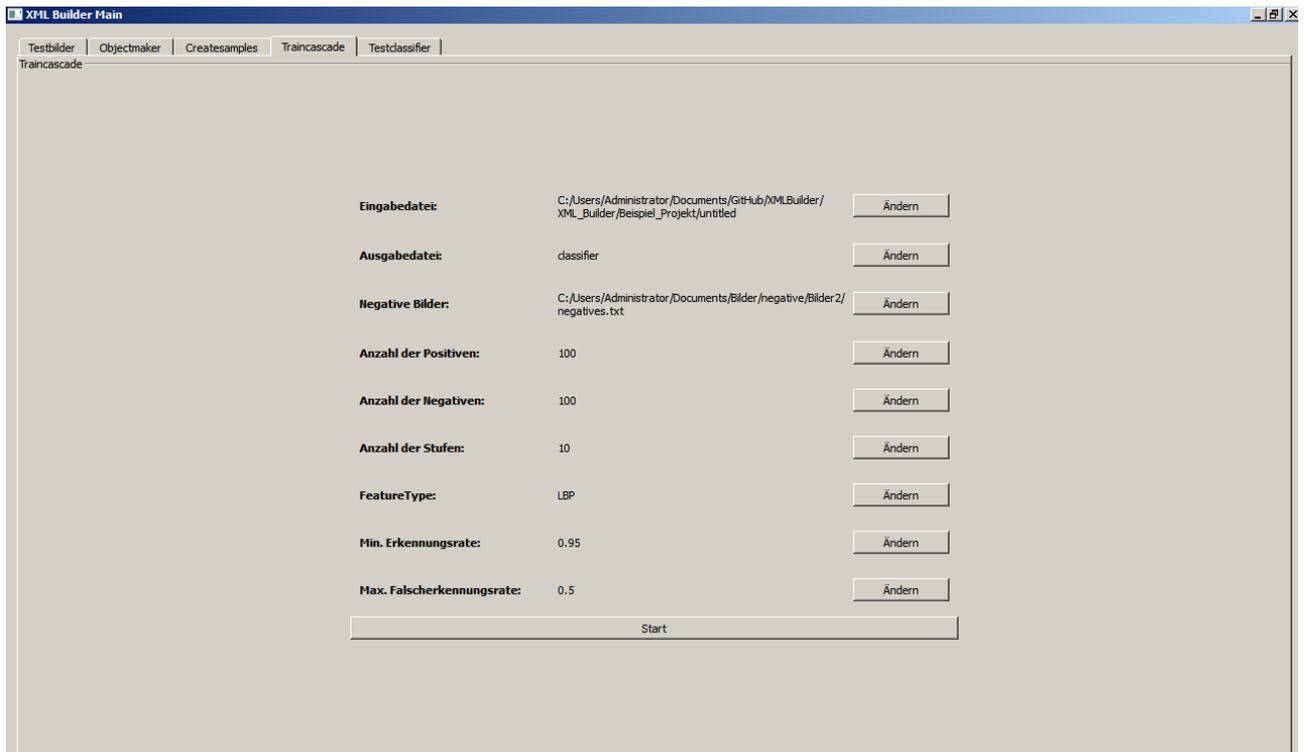


Abbildung 9: Beispiel für GUI-Traincascade

## Literatur- und Quellenverzeichnis

- [Ach13] ACHU: *Create Your Own Haar Classifier for Detecting objects in OpenCV.* <http://achuwilson.wordpress.com/2011/07/01/create-your-own-haar-classifier-for-detecting-objects-in-opencv/>.  
Version: 18.04.2013
- [Deg12] DEGENKOLBE, M.: Bachelorarbeit - Entwicklung eines Rahmenprogramms zur Spur- und Fahrzeugerkennung für mobile Endgeräte. (2012)
- [IIS13] IIS, Fraunhofer: *Detektion und Analyse von Objekten und Gesichtern.* <http://www.iis.fraunhofer.de/de/bf/bsy/fue/isyst/detektion.html>. Version: 18.04.2013
- [OPE13] OPENCV: *OpenCV 2.4.5.0 documentation.* [http://docs.opencv.org/doc/user\\_guide/ug\\_traincascade.html](http://docs.opencv.org/doc/user_guide/ug_traincascade.html). Version: 18.04.2013
- [VJ13] VIOLA, B. ; JONES, M.: *Rapid Object Detection using a Boosted Cascade of Simple Features.* [http://research.microsoft.com/en-us/um/people/viola/Pubs/Detect/violaJones\\_CVPR2001.pdf](http://research.microsoft.com/en-us/um/people/viola/Pubs/Detect/violaJones_CVPR2001.pdf). Version: 18.04.2013