Projekt 2 - Schneedetektion

* Repository: <https://github.com/uzapy/ch.bfh.bti7302.w2015.schneedetektion.git>
* Notizen: GoogleDocs

Projektplan

Logging

Unit-Tests

Dokumentation

Bilder

* Speicherort (Online / NAS / Share)
* Bilder pro Standort: DEZ 2014 / JAN 2015 / FEB 2015 / MAR 2014 / APR 2015
* Schema: **[Kamera]\_yyyyMMdd\_HHmmss** => 2 Stunden verschoben als Info im Bild
* Infos: 352x288 / RGB / JPEG / No Exif
* mvk021: Grauholz
* mvk101: Bern-West
* mvk105: Bern-Wankdorf
* mvk107: Bern-Ost
* mvk110: ???
* mvk120: Frutigen-Kandersteg?
* mvk122: Brienzersee Richtung Brienz?
* mvk131: Brienzersee Richtung Brienz?

Bild ohne Autos herstellen pro Standort

Bilder in **Regionen** unterteilen => Segmentierung

* Regionen-Kategorien (Bahn / Pannenstreifen / Rasen / Himmel / Hintergrund)
* Metadaten abspeichern => DB / File
* **Bitmasks**
* Fuzzy?

|  |  |
| --- | --- |
| **Ansatz 1** (Tag, keine Autos, kein Niederschlag, kein Nebel)   1. Existiert bereits: Referenz-Bild, Regionen pro Kamera-Bild, Durchschnitts-Sättigung pro Region 2. Bild laden 3. Durchschnittsfarbe pro Region berechnen 4. Sättigung pro Region mit Sättigung auf Orginal-Region vergleichen 5. Wenn eine Region positiv, dann positiv | https://upload.wikimedia.org/wikipedia/commons/e/ea/HSV_cone.png |

Playground-Modus

Live-Modus

Input: File Access => Service, der einen Ordner beobachtet

Input: Selber holen per WGET

Output: Webservice-Aufruf / Web / Push

Ja / Nein / Weiss nicht

Bessere Qualität => Dynamischere Regionen Berechnung

## Bild-Editor

1. .NET-Tool zum Bilder anzeigen und einfach bearbeiten
2. Polygon-Editor => Wie werden die Daten abgespeichert? => XML / JSON / **DB**?
   1. [Polygon-Editor](https://wpfshapedemo.codeplex.com/releases/view/18071) gefunden
   2. WPF-Tools mit dem [Extended.Wpf.Toolkit](https://www.nuget.org/packages/Extended.Wpf.Toolkit/)
   3. Serialisierung mit [Newtonsoft.Json](https://www.nuget.org/packages/newtonsoft.json/)
3. Masken-Editor
4. Zeitachse-Schieber
5. Polygon Editor mit OpenCV
   1. Masken Tool
   2. Bilder segmentieren per Polygonzug => Eventuell mit [OpenCV](http://www.emgu.com/wiki/index.php/Tutorial)
   3. [Emgu](http://www.emgu.com/wiki/index.php/Tutorial#Examples): C#-Wrapper für OpenCV
   4. Tutorial bei [OpenCV-Fehler mit EmguCV](http://www.codeproject.com/Articles/257502/Creating-Your-First-EMGU-Image-Processing-Project)

## Bilder einteilen in Kategorien

Refenz-Bilder finden, um die Vielschichtigkeit zu überblicken

Welche mögliche Zustände müssen abgedeckt werden => **Kategorien** finden (6-10)

* Tag
* Nacht
* Schnee
* Regen
* Nebel

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Standort | Bezeichnung | Bemerkungen |
| mvk021 | A1 Grauholz > Bern | A1 2.35 38-37 | 000 - A1 Ost |
| mvk101 | A1 Weyermannshaus > Grauholz | A1 161.5 34-35 | 011 - A1 West |
| mvk105 | A6 Wankdorf >Thun | A6 1.2 37-12 | A6 |
| mvk107 | A6 Ostring > Wankdorf | A6 3.0 12-37 | 025 - A6 Süd |
| mvk110 | A8 Gnoll (Brünig) > Luzern | A8 45.5 Gnoll | 037 - A8 |
| mvk120 | A8 Brienz > LU | A8 37.5 29-30 | 035 - A8 |
| mvk122 | A8 Faulensee > LU | A8 6.5 20-22 | 031 - A8 |
| mvk131 | A8 Soliwaldtunnel Süd > LU | A8 41.4 Soliwald | 036 - A8 |

Gefundene Kategorien => Abweichungen vom Referenzbild

* Tag / Nacht => Farbig / Schwarz-Weiss
* Nebel
* kein Niederschlag / Regen / Schnee
* Nasse Fahrbahn
* Lange Schatten
* Direkte Sonneneinstrahlung, Reflektion & Scheinwerferlicht

## Infos einfordern von Martin Lorenz / Daniel Bättig

Gibt es einen Webservice von Meteo-Schweiz mit Zusatz-Informationen (pro Koordinaten):

* Temperatur
* (Wetter)
* Datum
* Historische Daten
* Was kann man sonst noch aus den Meteo-Daten herausholen?

=> opendata.admin.ch => Sehr unverständlich; keine Ahnung, was da überhaupt wirklich angeboten wird...

Wetter: <http://opendata.admin.ch/de/dataset/messdatensmn>

Niederschlag: <http://opendata.admin.ch/de/dataset/messdatensmnprecip>

Open: <http://openweathermap.org/api>

JSON-Daten per GPS-Koordinaten abrufbar. 1200 Requests pro Minute gratis.

Request für Bern: <http://api.openweathermap.org/data/2.5/weather?lat=47&lon=8>

Request für Gruholz-Kamera: <http://api.openweathermap.org/data/2.5/weather?lat=46.9889&lon=7.4733&appid=bd82977b86bf27fb59a04b61b657fb6f>

## Bilder ohne Autos - Referenzbilder

Vergleich zwischen Referenzbild und Live-Bild => Hauptfrage: Gegen welche Referenz-Kategorie soll verglichen werden

Fahrbahn-Segment => Ist ein Auto drauf? Lässt sich das automatisiert herausfinden?

**Nachfragen bei Peter von Niederhäusern**: Laufen die Kameras aktuell? => Ja!

## Zugriff auf den Server mit dem Cron-Job

Bei Peter von Niederhäusern, ob der Bilder-Herunterlade-Job aktuell läuft.

Peter von Niederhäusern: *Damit wir das möglichst einfach halten können, werde ich Dir einen*

*lokalen Account auf dem Server erstellen (z. B. Dein BFH Kürzel).*

*Dann werde ich den Cronjob in Deiner cron list eintragen.*

*Könntest Du mir dazu (D)einen public key zusenden, damit das mit dem*

*Login einfacher geht?*

Keypair generiert:

---- BEGIN SSH2 PUBLIC KEY ----

Comment: "rsa-key-20151010"

AAAAB3NzaC1yc2EAAAABJQAAAQEAoBj9InVeVab3irxzh/Yey5VihNqZINg60t+V

whqP+5sUv9oAHyqBRHCQZFXjnApo/jLZfFkKd7MWkvlnRFPeR0JYoSOYmlyz8urh

1IJgzQ1fNhl6j0lIWmDGyzq6VdlFOx6c9iPQ/dPrNH8Rkt31x0ARUWrBIn/nRDsw

Bx/AJuyAnnYwvBcyTCCA5N/vMOEH4evyfUyZt+VVkT+Vx5ZvVgcaFYLdzu43BlRP

5p+8byOrZB5ddQnQdcXCqYpwf6nCZ/pWwr3g0ZAbi5IGti0pffkhx6FCu5Yk9Ogq

ZHFp3fp458uPjmmJAhW07zdm8ZZwO8MIjl7hr61K8Su35N3nWw==

---- END SSH2 PUBLIC KEY ----

Einrichten mit Peter von Niederhäusern

* Key neu generiert auf MacBook. Oberer Key wurde nicht gebraucht
* Der neu generierte Key ist gespeichert in: Macintosh HD/users/uzapy/.ssh
* Verbinden mit Athena: > ssh [bublm1@athena.bfh.ch](mailto:bublm1@athena.bfh.ch)
* Verbose login: > ssh -vvv -i id\_rsa [bublm1@athena.bfh.ch](mailto:bublm1@athena.bfh.ch)
* Zugriff auf CronTab: > crontab -e
* SSH Verlassen: > exit

## Zeitdiskrepanzen analysieren

Bilder haben einen anderen Timestamp als als im Datei-Namen. Mit dem Datum und der genauen Sonnenuntergangs- und Aufgangs-Zeit liesse sich der effektive Zeit herausfinden.

Vermutung: Ein Teil der Informationen sind nicht in MEZ sondern in CET oder die Winterzeit wurde bei der Aufnahme nicht beachtet.

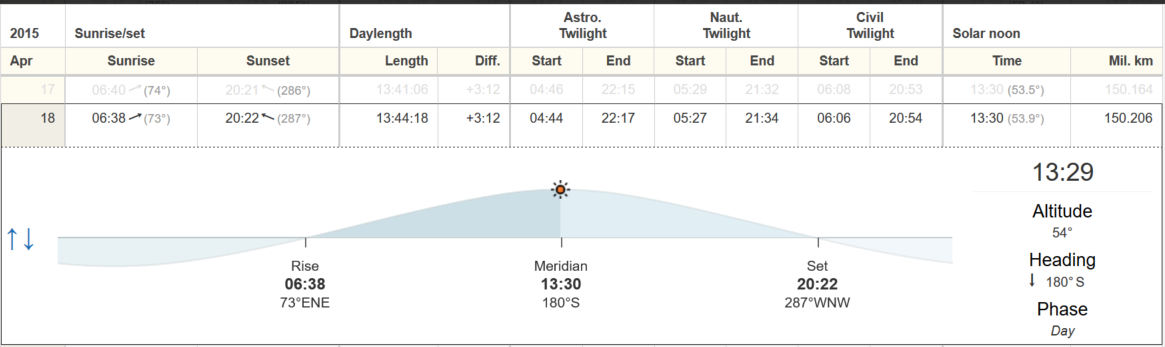
Lorenz Martin: *Meist werden die Wetterdaten in UTC (Weltzeit) abgespeichert, d.h. Greenwich-Zeit OHNE Sommerzeit. Möglicherweise ist das auch bei den Kameradaten der Fall: Die im Bild eingeblendete Zeit ist UTC, die Zeit im Dateinamen ist lokale Zeit (Zeit auf dem Server, wo die Bilder per Skript abgelegt werden).*

Habe mir die Bilder noch einmal angeschaut. Mir ist vorher nicht aufgefallen, dass im Timestamp zwischen Datum und Zeit jeweils CET und CEST steht. Wie es aussieht stimmt die Zeit im Bild, während die Zeit im Dateinamen nicht korrekt ist. Sichtbar ist es, wenn man den Sonnenuntergang im Wankdorf mit den historischen Daten vergleicht.



|  |  |  |
| --- | --- | --- |
| Bild | 2015-04-18 CEST 20:30:56 | 20:30 |
| Dateiname | mvk101\_20150418\_183002 | 18:30 |

Laut [timeanddtate.com](http://www.timeanddate.com/sun/switzerland/bern?month=4&year=2015) war der Sonnenuntergang in Bern am 18. April 2015 um 20:22. Der Himmel und die Helligkeit im Bild entsprechen eher 20:30 als 18:30. Wäre es 18:30 wäre der Himmel heller und blauer.



## Nachgefragt

Bei Daniel Bättig, ob es einen Service gäbe, der historische Wetterdaten liefern kann.

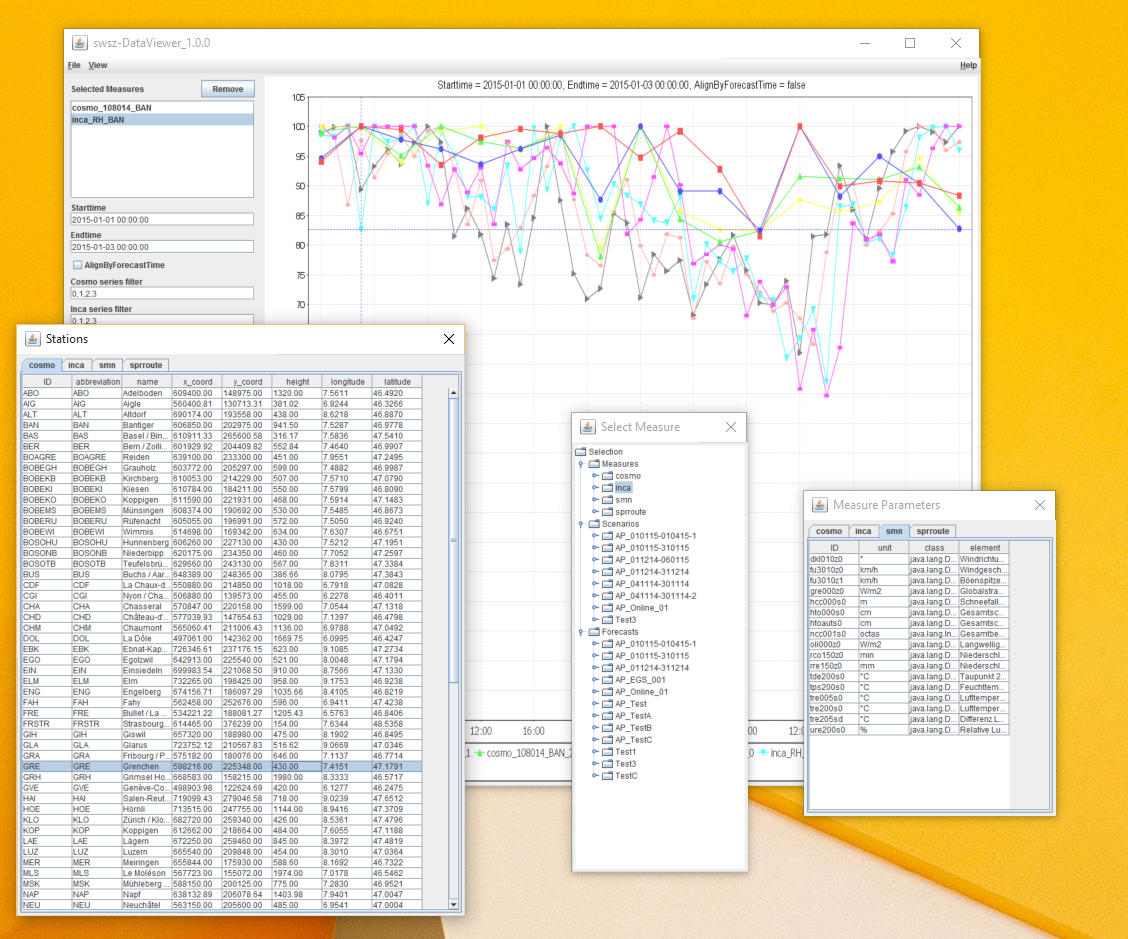
=> Meteodaten zu jedem Bild herunterladen und abspeichern

Daniel Bättig: *Wir haben einen Data-Viewer, bei dem Sie Sonden- und Wetterdaten der MeteoCH abrufen können. Sie finden den Daten-Viewer – eine Java-Applikation – beigelegt. Sie müssen sich dann per VPN auf das BFH-Netz einloggen, damit das Programm auf die Datenbank der Daten zugreifen kann. In der nächsten Woche (KW42) wird aber die Datenbank nicht zur Verfügung sein, da wir für den nächsten Winter eine Neu-konfiguration der Datenbank aufsetzten.*

Jar-Datei zum Laufen gebracht per Konsole:

> "C:\Program Files\Java\jre1.8.0\_60\bin\javaw.exe" -jar C:\Projekte\_GitHub\ch.bfh.bti7302.w2015.schneedetektion\Meta\swsz-dataviewer-0.0.1-SNAPSHOT\_v4.jar

Jetzt müsste man nur noch wissen wie das Ding funktioniert und was man damit eigentlich machen kann… => Bei Daniel Bättig nachgefragt.



Variante um sich direkt an eine MySQL Datenbank zu [verbinden](http://www.codeproject.com/Tips/423233/How-to-Connect-to-MySQL-Using-Csharp) mit C#.

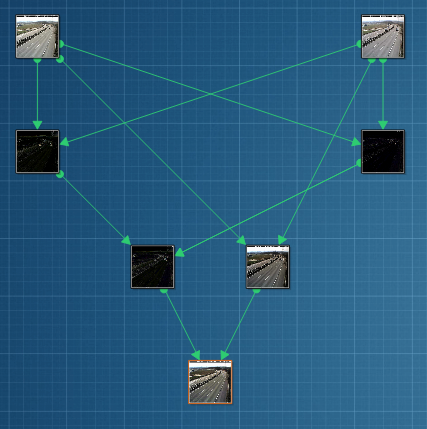
## Genaue Position der Standorte

|  |  |  |
| --- | --- | --- |
| Name | Standort | Koordinaten |
| mvk021 | A1 Grauholz > Bern | 46°59'20.1"N 7°28'24.0"E |
| mvk101 | A1 Weyermannshaus > Grauholz | 46°57'02.9"N 7°24'29.2"E |
| mvk105 | A6 Wankdorf >Thun | 46°57'58.2"N 7°28'15.0"E |
| mvk107 | A6 Ostring > Wankdorf | (46°57'04.7"N 7°28'19.5"E) |
| mvk110 | A8 Gnoll (Brünig) > Luzern | 46°45'05.5"N 8°07'42.0"E |
| mvk120 | A8 Brienz > LU | 46°44'33.3"N 8°03'46.0"E |
| mvk122 | A8 Faulensee > LU | 46°40'04.6"N 7°43'20.5"E |
| mvk131 | A8 Soliwaldtunnel Süd > LU | 46°45'05.5"N 8°06'29.8"E |



## Bilder ohne Autos

Habe etwas mit ImagePlay herumgespielt. Es scheint, als ob die Bilder zu viele JPEG-Artefakte haben, dass man die Autos auf der Strasse mit eigen Arithmetischen Operation wegoperieren könnte.

## Besprechung 16. Okt 2015

Tool

* Überblick
* Time Lapse
* Polygon-Tool => Masken anwenden => Segmentierung

Meta-Daten zu den Bildern

* **DONE**: Datum & Zeit im File
* **DONE**: GPS
* Meteodaten
  + DataViewer von Daniel Bättig
  + <http://openweathermap.org/api>
  + Was genau und wo ist die DB
  + Vorbeigehen bei Daniel Bättig kennenlernen, zeigen

Referenzbilder pro Kategorie

**DONE**: Kategorien

Aktuelle Bilder

* Peter von Niederhäusern
* Cron-Job läuft
* Warte noch auf Zugang

GitHub

### Resultate der Besprechung

* **DONE**: Besuch bei Daniel Bättig organisieren => Donnerstag, 29. Oktober 2015 15:55 in Burgdorf
  + Wetterdaten auf Server: **dataviewer\:dvuser@sgps-1.bfh.ch/swsz\_14\_15**
* Mehr Kameras einbinden und eventuell häufiger Bilder herunterladen. Dabei an den Disk-Speicher denken. Eventuell könnte beim Cron-Job auch ein Preprocessing passieren
* Alle 10 Sekunden Bild machen und Autos bereinigen - Vorverarbeitungs-Job
* Segmentierung / **DONE**: Maskierung mit OpenCV
  + **DONE**: Polygone abspeichern mit Punkten zwischen 0 und 1
  + Nur Bounding-Box weiterverarbeiten
  + Mehrere Masken kombinieren => Wahrscheinlich nicht relevant
* Autos bereinigen
  + pro Stunde
  + pro Kategorie
  + kontinueirlich (automatischer Autobereinigungs-Algorithmus bereits im Cron-Job)
* **DONE**. Korrekte Daten aus den Bild-Dateien herauslesen (Datum, Zeit und Breite und Höhe)
* Wie viele Events haben wir eigentlich? Kontroll-Bilder erstellen
* Verarbeitungszeit => können wir alles Live machen? Was muss schliesslich heruntergeschrieben werden?

[Icons](https://www.iconfinder.com/iconsets/linecon)

## Besprechung 13. Nov 2015

Halbzeit. 3 Fronten

Zugang zu Wetterdaten

* Herunterladen für die Spielwiese
* Automatisch einbinden live
* **Problem**: Nachbauen es Clients

Zugang zum Download der Bilder

* Häufiger Bild machen zum Autos Bereinigen
* (Pre-) Processing direkt auf Athena
* **Problem**: Wenig Ahnung von Linux und Shell Scripts
* **Priorität**: Server-Job: Alle x Sekunden Bild machen => Autobereinigungs Job
  + Neuer CronJob: Jeden Tag im November von 12:00 bis 13:00 jede Minute ein Bild herunterladen und in neues Verzeichnis ablegen
    - Verzeichnis: /srv/athena.bfh.ch/projects/astra\_burst/
    - CronTab: \* 12 \* 11 \* ~/bin/astraburst.sh
* Kamera-Server: http://www.astramobcam.ch/benutzer/login.php

Naiver Algorithmus

* Vergleich mit Referenzbildern
* Trainieren mit Kontrollbildern
* **Problem**: OpenCV noch nicht verstanden
* **Priorität**: Autos eliminieren
  + Mehrere Bilder schnell nacheinander
  + Bilder ohne Autos
* Kategorien einbeziehen (Monat / Woche / Stunde / Wetter)
  + Kategorien-Bilder ohne Autos herstellen
  + Grundsätzlich schwieriger: Dämmerung / Morgengrauen
* Events markieren => Kontrollbilder

Nächste Sitzung 18. Dez 2015 im Rolex

## Besprechung 18. Dez 2015

Zugang zum Download der Bilder

* **Server-Job**: Alle **10** Sekunden Bild machen => Autobereinigungs Job
* **Neuer CronJob**: Jeden Tag im November von 12:00 bis 13:00 jede Minute ein Bild herunterladen und in neues Verzeichnis ablegen
  + Verzeichnis: /srv/athena.bfh.ch/projects/astra\_burst/
  + CronTab: \* 12 \* 11 \* ~/bin/astraburst.sh

Naiver Algorithmus

* OpenCV einigermassen verstanden
* **Autos eliminieren**
  + Mehrere Bilder schnell nacheinander
  + Bilder ohne Autos
* Kategorien einbeziehen (Monat / Woche / Stunde / Wetter)
  + Kategorien-Bilder ohne Autos herstellen
  + Grundsätzlich schwieriger: Dämmerung / Morgengrauen
* Events markieren => Kontrollbilder

Zugang zu Wetterdaten

* No Update

Jetzt: **Zusammenbauen!**

## Abschlussbesprechung 03. Februar 2016

* Algorithmus zusammengebaut
* Geschwindigkeit mit dem aktuellen Algorithmus kein Thema, Alle 10s ein neues Bild
* Detektion noch nicht gut genug