Automatic Semantic Labelling of Urban Areas using a rule-based approach and realized with MeVisLab

Thijmen Speldekamp, Chris Fries, Caroline Gevaert, Markus Gerke
04-2015

Abstract

The article is about a project done at the ITC faculty of the University of Twente. It describes a rule-based system and the actual implementation of a model to perform an automatic semantic labelling of urban areas. There will be a short introduction and some information about the software that was used in this project and the subset data that was used. Followed by the explanation of the model we will show and assess the result and the accuracy assessment.

Online version and MeVisLab script available at
https://www.researchgate.net/publication/275639040_Automatic_Semantic_Labelling_of_Urban_Areas_using_a_rule-based_approach_and_realized_with_MeVisLab

Introduction

This report is the result of the individual final assignment from the Geo Data Processing & Spatial Information given at the ITC faculty of the University of Twente. For this assignment we looked at automatic semantic labelling of very high resolution airborne images (derived ortho image and height model). The assignment consisted of making a model to perform this automatic semantic labelling in a simple program.

Method

MeVisLab

For this project a program called MeVisLab (http://www.mevislab.de/) was used. This is a program originally intended for medical image processing and visualization. This means that the program is not used for its original purpose, which brings new insights for the subject, but also problems.

The helpguide in Mevislab doesn't contain much explanation concerning problems you encounter when producing a model for automatic semantic labelling.

The program uses modules, which allows for a simplified way of programming. The modules are connected to each other to extract information from one another. The network that is created is the model which in this case performed the automatic semantic labelling.

Subsets

For the project the TOP (True Ortho Photos) & DSM (Digital Surface Models) files that was provided by the ISPRS (http://www2.isprs.org/commissions/comm3/wg4/semantic-labeling.html) were used. The 'area 30' subset, created by the ISPRS, was used as the training sample, because this was a subset which had a good division between the four bigger classes: impervious surfaces, buildings, trees & low vegetation. It also allowed for experimenting with the car class, since this is the most challenging.
We also made use of the 'area 26' subset to set some parameters in the training sample, the reason therefore was that there is water present in this subset. For the testing stage, the 'area 32' subset was used.

The Model:

Image processing

The TOP file is loaded into the model via an image load operator and hereafter dissected in three bands, green, red, near infrared. Hereafter two new arithmetic modules are placed. The first one is to calculate the NDVI values of the image. The formula used in this arithmetic is:

$$NDVI = \frac{NIR - Red}{NIR + Red + 0{,}0001}$$

In the formula NIR stands for near infrared. The term "+ 0,0001" in the denominator is used to prevent errors in the data because of dividing by 0.

The second arithmetic is to calculate the Intensity. This will be used to find shadows and very dark areas in the image. The formula used here is a simple average of the three input images. This data is then sent to the thresholds, which will be discussed later.

The use of the NDVI is based on the fact that green vegetation has a low reflectance in the red region due to chlorophyll and high reflectance in NIR due to the cell structure. The reflectance in the NIR is much higher than in the red region. This is a unique characteristic of vegetation.

So when you calculate the NDVI, if NIR is much higher than red, you will get a value closer to 1. This is why green vegetation has higher NDVI values and all other objects such as cars, roads, etc. have low NDVI values.

Another benefit is that it looks at the relative difference between NIR and red, so it is also capable of identifying vegetation in shadows. This is shown in figure 2. The same counts for the low vegetation in the image.

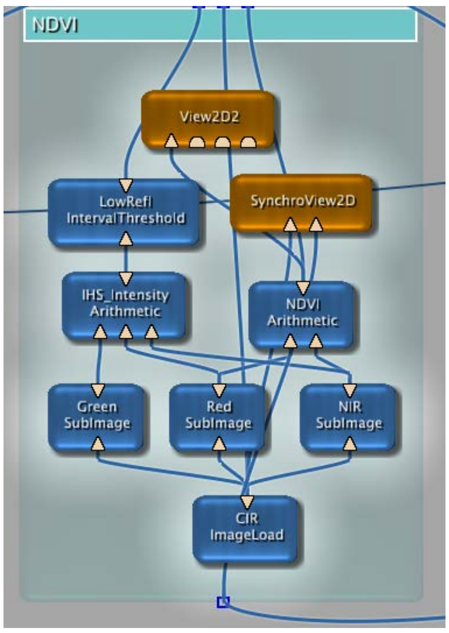


Figure 1: The modules used for the NDVI

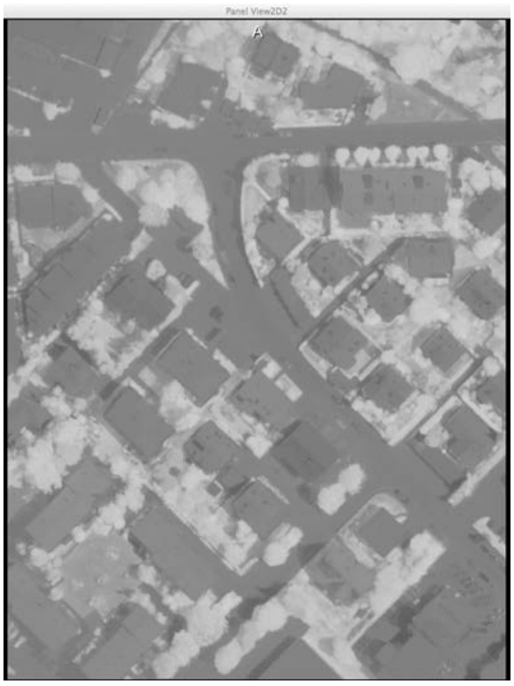


Figure 2: The image from the NDVI output

The DSM file is also loaded via an image load operator. The grey values in this file are encoded as 32 bit float values. But the DSM files in the data set contain values, which don't represent a surface height. For instance if a slope is present inside an image a building on one side of the image may be lower (in height) than a road on the other side. For an automatic sematic labelling we need to adjust this.

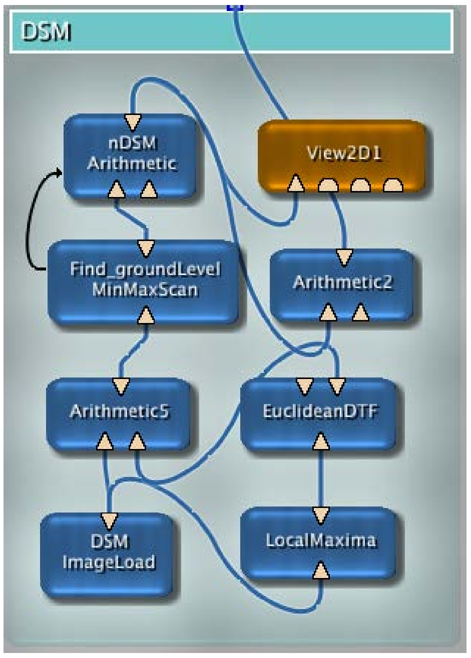


Figure 3: The modules used for the DSM

In our model we used normalized DSM data. Using the lastools-toolbox (http://rapidlasso.com/lastools/), the DSM image is divided into ground and off-ground pixels. Then for all off-ground pixels the closest ground point is assumed to be the relevant low point. The height is then calculated by the subtraction of the height from the closest ground pixel from the off ground pixel its assigned to. Using this method the normalized DSM's (nDSM) created gave better values when working with threshold segmentation of the classes. The output from the DSM group is also linked to the thresholds[1].

[1] The nDSM tiles are available at http://www.researchgate.net/profile/Markus_Gerke/publication/270450634_normalized_DSM_heights_encoded_in_dm_see_report/links/54aaaa

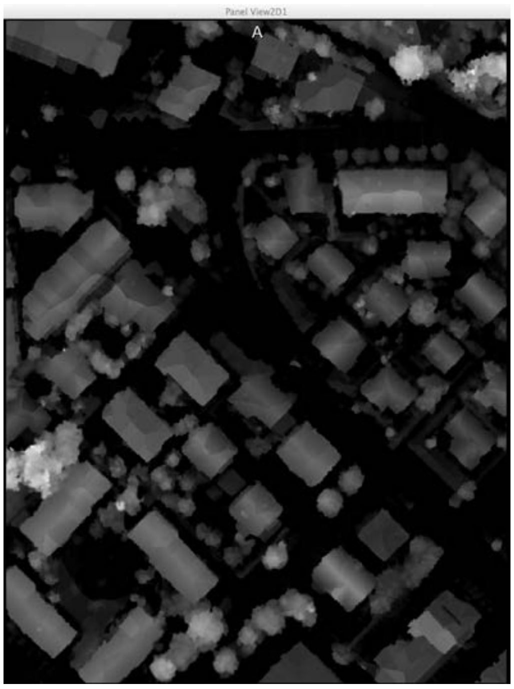


Figure 4: The image of the normalized DSM, which is used in the final model.

Shadows

Using a Euclidian distance module and then a threshold for this distance an area around buildings was captured. Then using an interval threshold that separated the low reflectance areas of the image together with the area around the buildings the shadows of buildings were separated. We did the same thing for a somewhat smaller area around trees and added these shadows to the CastShadow arithmetic. We thus captured all the shadows of the image. We saw in our attempts to segment the different classes that shadows were classified incorrectly. We decided to separate the shadows on high areas from low areas using another height threshold. Then we used masks of our road segment and masks of our low vegetation segment to separate shadows that were on roads and shadows that were on low vegetation. All these modules produced three different outputs from the shadow group. One for the higher shadows (on buildings), one for the lower shadows on roads and one for the lower shadows on vegetation.

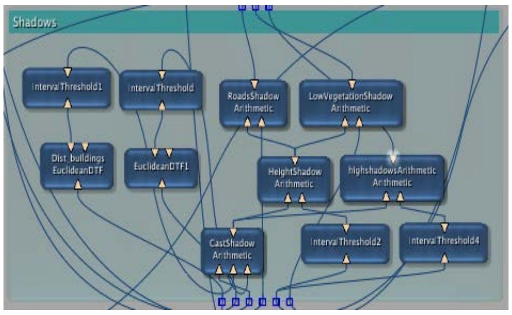


Figure 5: The modules for the shadow separation.

Classes

The different classes were set up by a certain amount of modules per class. In most classes some of these modules are the same, but the parameters used in them were different since the classes had their own features. Each class will now be discussed.

Roads (or impervious surfaces)

The data from the NDVI and the DSM are first processed by two threshold modules, which exclude the all the impervious surfaces in the subset. The values used for this exclusion are as follows:
NDVI threshold: -0.15 to 0.1
DSM threshold: 0 to 12
Hereafter these outputs are combined in an arithmetic module. This arithmetic module states: (a or (b and c)) and !d.
These variables stand for the inputs in the arithmetic module, where ‘a’ the output from the RoadShadows arithmetic is. ‘b’ the output from the NDVI threshold, ‘c‘ the output from the DSM threshold and ‘d’ the output

bc0cf2ce2df668aac3?origin=publication_detail

from the HighShadows arithmetic output.
So not only the outcome from the thresholds are taken into account, the shadows on the roads are also included and shadows that lie on the low vegetation are not taken into account.
From the Arithmetic module the data goes to the morphology modules. There are two of these modules following each other. The first module will perform a dilation filter and the second one an erosion filter. This thus preforms a closing morphology of the road class.

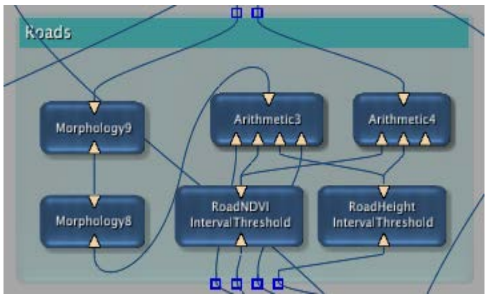
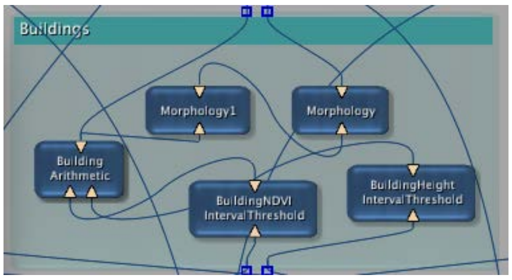


Figure 7: The modules used for the impervious surface class.

Buildings

The data from the NDVI and the DSM comes in by the thresholds; which has the following parameters:
NDVI threshold: -0.1 to 0.3
DSM threshold: 12 to MaxHeight
The arithmetic is here less complicated. In the building arithmetic the output from the NDVI and the DSM are both taken into account, and put through to the morphology filters. In the case of the buildings the morphology filters are placed the other way around as which was the case with the impervious surfaces, so first the erosion filter and then the dilation filter. This causes an opening morphology and we did this so buildings would have less "sharp edges" to the area around it. We did an opening at the buildings and a closing on the roads because it would then be smoothed out the best in our Interval filter we will address later. The parameters on the filters are set to higher values, 18 pixel size kernel, because buildings have sharper edges. These values will be later explained in the morphology section.


Buildings
Morphology1
Morphology
Building Arithmetic
BuildingNDVI IntervalThreshold
BuildingHeight IntervalThreshold
Figure 8: The modules used for the buildings class.

Low vegetation
With the low vegetation the setup is more similar to the impervious surfaces setup, this is because with the low vegetation the shadows are again taken in to account with the classification. But it starts the same with the NDVI and DSM data going in the threshold, which uses the following parameters.
NDVI threshold: 0.1 to 0.6
DSM threshold: 0 to 10
The following arithmetic states: a or (b and c).
Which means that the shadows on the low vegetation (a) are taken into account. It further takes into account the NDVI (b) and the DSM (c).
With the morphology filters the data is first dilated and then eroded. Although the filters are there, it is set to a lower value, 3, since low vegetation has few sharp edges.

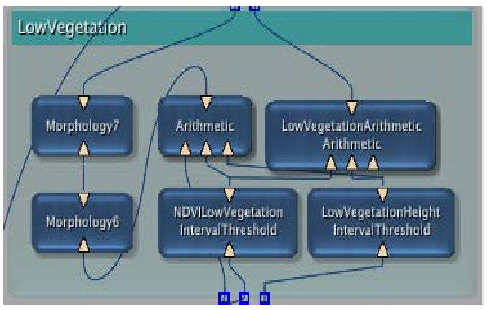

LowVegetation
Morphology7
Arithmetic
LowVegetationArithmetic Arithmetic
Morphology6
NDVILowVegetation IntervalThreshold
LowVegetationHeight IntervalThreshold

Figure 9: The modules used for the low vegetation class.

Trees
The setup for the tree class is again similar to that one of the buildings. With the following parameters in the thresholds:
NDVI threshold: 0.3 to 1
DSM threshold: 10 to MaxHeight
In the arithmetic both outputs from the thresholds are put together and send to the morphology filters. These filters are set to a lower value, 9 for both filters. For the same reason as for the low vegetation filters, this is because the area that are filtered away by the tree class, has to be in most cases by assigned to the low vegetation class.

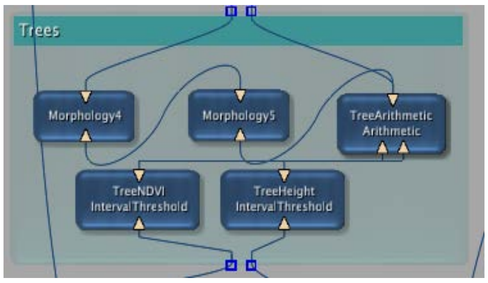

Trees
Morphology4
Morphology5
TreeArithmetic Arithmetic
TreeNDVI IntervalThreshold
TreeHeight IntervalThreshold

Figure 10: The modules used for the tree class.

Cars
The car class was the most difficult one, since cars have a low area size, not much height difference to roads and have different NDVIs.

So the NDVI and height interval thresholds were pretty wide to at least capture all the cars. The problem now was that the car class would wrongly assign a lot of pixels which belonged to other classes to the car class. To counter this problem we implemented a high reflectance threshold. The really high reflecting parts of the image were classified as cars using a threshold taking input of the intensity image.

The height and NDVI thresholds are as follows:
NDVI threshold: 0 to 0.3
DSM threshold: 0 to 18
This is then again sent to the morphology filters. In the order erosion and dilation which as in the building class will perform an opening morphology. The parameters set on these filters are low; cars themselves aren't large area objects.

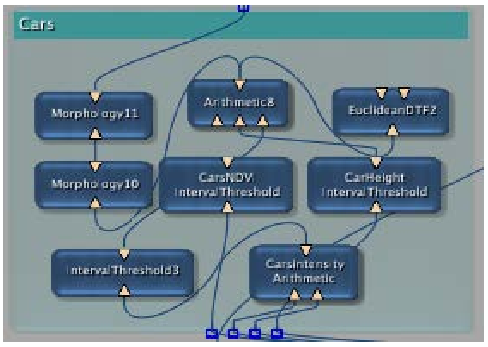


Figure 11: The modules used for the car class.

Clutter/background
The clutter class consists of everything that isn't included in the previous mentioned classes.
Also the clutter class setup is similar to the others, but in this case the height is not taken in to account, because this class has to pick up the spots that were not classified by the other classes. This also means that the parameters on the morphology filters are low.
Furthermore water is also included in this class.

Nevertheless the parameters in the thresholds were set to the following values, of which the DSM can be ignored.
NDVI threshold: -1 to 0
In the following arithmetic module there is the shadows on the roads and on the low vegetation are not included in the clutter class. This is done by the statement in the arithmetic module; a and !b and !c.

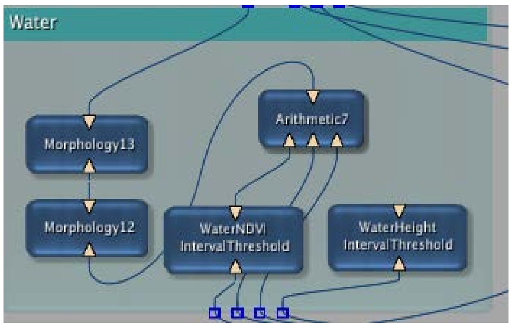


Figure 12: The modules used for the clutter/background class.

Morphology
With the morphology filters the edges of the projection of the classes will be smoothened. To do this, there are two Morphology modules needed, one that does the erosion filtering and the second one does the dilation filtering. In the module the x and y kernel can be altered to get different results. This gives a region to where the filtering is applied. These are the values that are mentioned in the classes. In the project the values of the x and y kernel were set to give the best result.

In the two images below area 30 is shown, first without morphology filtering on the building, see figure 13, and then after applying the morphology filter, see figure 14.

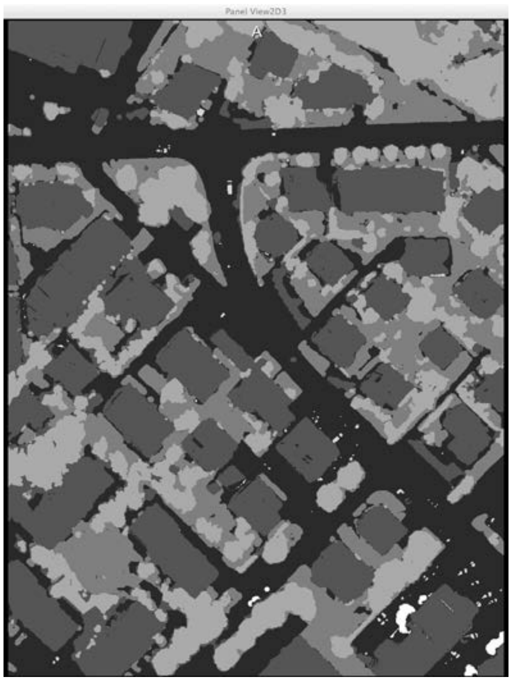


Figure 13: Image of the buildings of area 30 before morphology filters applied

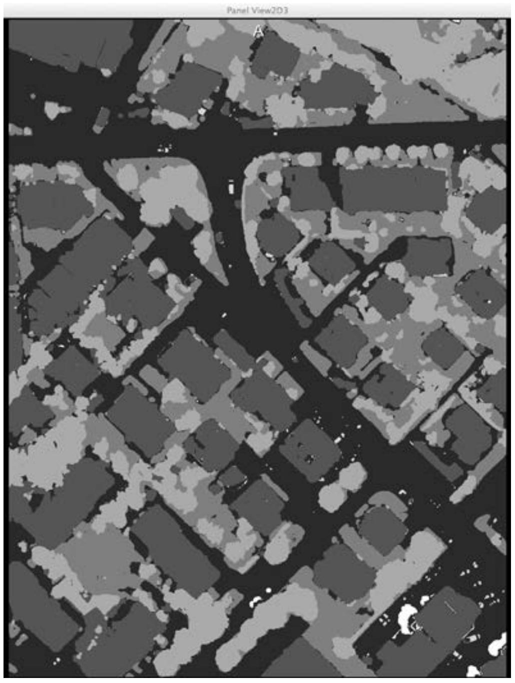


Figure 14: Image of the buildings of area 30 with morphology filters applied

Displaying the classification
At the stage were we defined our requirements for a pixel to be in a class we created six different bit images in which a pixel, which meets the requirements, gets a value of 1, if not, a value of 0. So we have six different images. We want an image

in which each pixel is assigned a class and in this image we should be able to see the different classes in different colours.

To make the raster image, six more thresholds were added to the model, one for each class. In these thresholds the statement was made that when the input is 1, the output value of the new thresholds will be assigned a value. For the different classes these values are:

Impervious Surface = 1
Buildings = 2
Low Vegetation = 3
Trees = 4
Cars = 5
Clutter/Background = 6

These outputs were then connected to an arithmetic module, in which the class numbers are added to each other. This output gives an image of the six classes. But for our automatic sematic labelling we need each pixel to only represent one class. When we add our six images we then have a problem when two or more of these six input images overlap. When that occurs, two or more classes are added giving a value higher than six, which doesn't represent a class or a value within the interval of (0 – 6). It could also occur that values were in the interval (0 – 6) but was not a right class value for that pixel. For example, when one pixel has the value of Impervious Surface added to the value of Buildings it would appear as low vegetation because 1 + 2 = 3.

To solve this, we first needed to filter out pixels, which had multiple values added to each other. To do this we adjusted our arithmetic to the following:

$$(a + b + c + d + e + f) - (10 * ((a * b) + (a * c) + (a * d) + (a * e) + (b * c) + (b * d) + (b * e) + (b * f) + (c * d) + (c * e) + (c * f) + (d * e) + (d * f) + (e * f)))$$

The first row is just the addition of all the images. What happens in the next rows is that when an image is multiplied, it gets a new value on that point and is then subtracted from the total image after being multiplied by 10. So what happens is that all the raster points which overlap now become a negative number.

The working of the above mentioned formula is shown in table 2. Because the middle pixel is double assigned, the following happens. The 1 multiplied by 2, is given the value -2, this is then done excluded from the added table after being multiplied by 10, which gives a result of 2 – 20 = -18.

1	0	1		0	2	0		0	0	0
0	1	0	*	2	2	2	=	0	-18	0
1	0	1		0	2	0		0	0	0

Table 1: example for the formula used for filtering out the double assigned pixels.

To filter out these double assigned pixels an IntervalMap module was added. In this module the following statements were made:
First there was stated that there could be only six different classes, after that the numbers below zero that occurred in the image were assigned to one of the six classes. For example, when something has the value -118, it could be an addition of 6 and 3, being clutter and low

vegetation respectively, this value is then assigned the value 3, because it is more likely to be low vegetation than clutter.

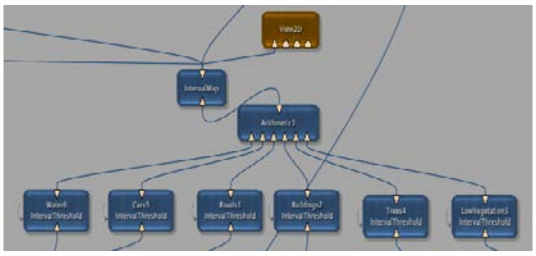


Figure 15: The modules needed for visualising the six classes in a raster.

Results

Some example results from the classification of the training and the test subset are found in figure 16 &17. The colour class figuration is according to the benchmark requirements:

White: impervious surfaces
Blue: buildings
Light Blue: low vegetation
Green: trees
Yellow: cars
Red: clutter

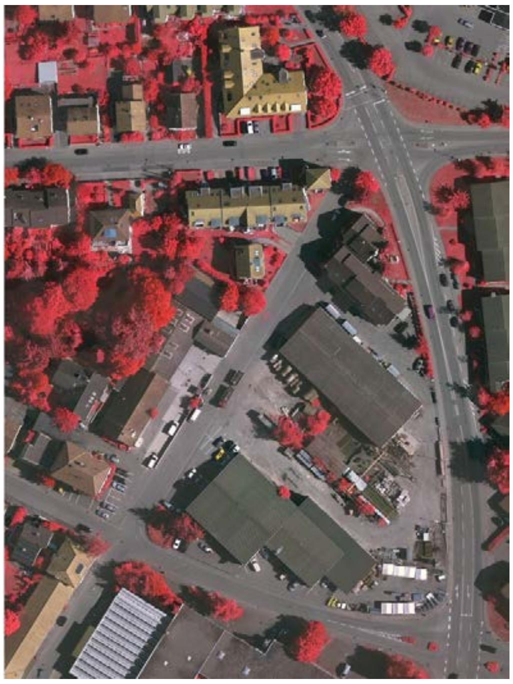


Figure 16: TOP area 29

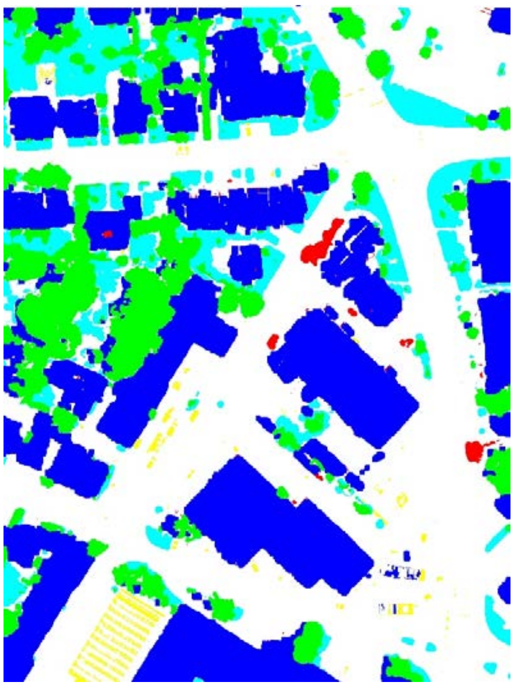


Figure 17: Classification result area 29

Accuracy assessment

The results on the ISPRS website were assed and resulted in an overall accuracy, on all validation subsets, of 81.8%. Looking at the individual classes the problem cases are identified as: cars have an average accuracy of 9%, most cars

are missed and labelled as impervious surface, in turn some buildings got labelled as car entirely, which is due to problems with the DSM normalisation.
Besides we observe wrong classifications in shadowed areas.

Conclusion

The conclusion from this project is that the automatic semantic labelling works pretty well, also given that the Mevislab software is not meant for this kind of application. The classes, which have a larger areas like impervious surfaces, buildings, low vegetation and trees do rather well. Compared to other results shown so far for the benchmark, especially in case of “car”, our approach underperforms dramatically. Many cars are not represented in the nDSM at all (either they are not in the DSM, or labelled as ground points during terrain filtering), and our approach relies on a certain height. Compared to the supervised approaches where also the appearance is learnt, we did not exploit the radiometric appearance and this is the major shortfall.

The mevislab script is found in the ResearchGate entry attached to this report (profile of Markus Gerke).