Finding an optimal dissimilarity measure for hierarchical segmentation of satellite images using alpha-trees

Jeroen Lammers

Abstract— In this paper we will introduce a new method of computing the dissimilarity between pixels based on a combination of the forward difference, backward difference and central difference. We will compare this method against the L_p-norm dissimilarity measure which is augmented using edge detector and ridge detector signals. We will also introduce an approximation of the area score which will be used to determine the quality of the results. Based on the resulting training scores we can conclude that the newly proposed method preforms slightly worse compared to the other methods. All methods provide robust solution when considering the scores on similar input data. However, due to problems which most likely stem from the area score approximation we obtain over-merged results from our training process which raises questions about the usability of the results.

Index Terms—Alpha-tree; Gabor filters; Difference filters; Horizontal cut; Optimal filters; Baysian optimisation

1 Introduction

Over the years the importance of satellite images has increased and as the technology progresses we find ourselves with more and more data. From this arises the need for automated solutions to segment these images that can cluster relevant data. It is however not well defined what the relevant data in such an image might be. For example, some might be interested in each specific house in a street while others might care for the segmentation of districts within a city. Ideally we would like to find a representation of our image such that we can easily segment different levels of detail in our image. This is an example of hierarchical segmentation. In this paper we will look at one such hierarchical segmentation method for satellite images, namely segmentation based on α-trees [9, 12].

We will be augmenting the L_p-norm dissimilarity measure with additional data such as the signals of edge and ridge detectors. This data will be extracted using banks of Gabor filters [20, 21] and banks of central difference derivative approximations and 1D Laplacian filters. We will also introduce a new dissimilarity measure based on the forward difference, backward difference and the central difference around a pixel. We will train these filters on single images with multiple ground truths in order to find suitable parameters. This paper will compare the performance of the resulting filters after training. We will look at the robustness of the filters by applying them to similar data and comparing the resulting performance. Lastly, we will perform a visual comparison between the resulting segmentations.

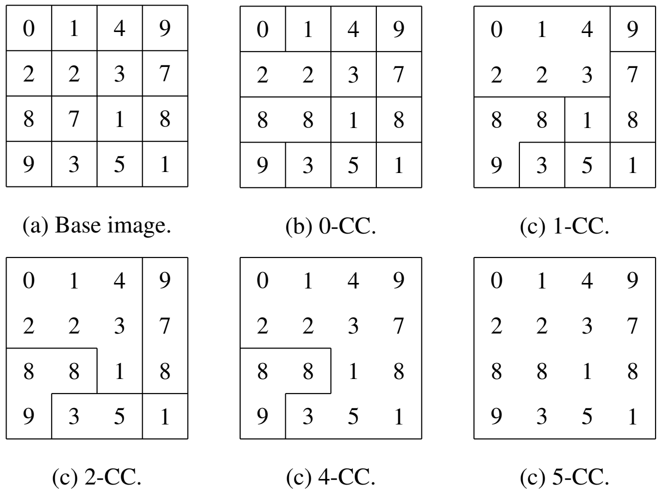


Fig. 1. An input image with its corresponding α-connected components using the absolute difference as a dissimilarity measure with 4-connectivity.

2 Preliminaries

This section will provide the reader with all the information needed to understand the later sections of the paper. We will start by providing an overview of α-trees, then we will consider all the filters that will be applied on our images as well as the post-processing done on the signals generated from the filters. Next we will go into more detail on how we define an α-tree to have a good segmentation of a ground truth image. Lastly we will look at the optimisation method used for the training phase.

2.1 Alpha-trees

In order to construct an α-tree of an image we first need to define a dissimilarity between pixels. One of the simplest dissimilarity measures is the L_p norm between neighbouring pixels. We define an α-connected component to be a set of pixels in which any pixel in the set can be connected to any other pixel in the set by a path between pixels in which every edge in the path has a dissimilarity less than or equal to α. An example of this is provided in figure 1.

We can represent the α-connected components in a tree. The leaves are defined as the 0-connected components. As the value of α increases, the connected components merge into larger connected components. The merged connected components are represented by new nodes in the tree with the corresponding α value. The root node is defined to be the node at which point there only consists a single connected component. In our example this will be at an α of 5. The α-tree corresponding to the example in figure 1 is represented in figure 2.

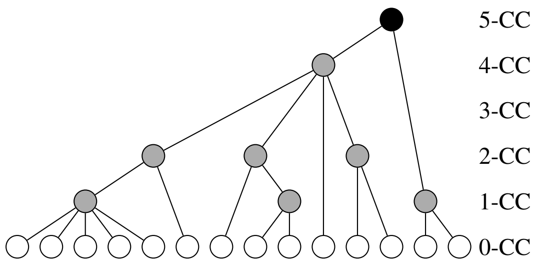


Fig. 2. The α-tree corresponding to the connected components of figure 1

In order to retrieve information from the α-tree we can apply a horizontal cut. A horizontal cut is computed by merging the leaves of the α-tree to a given level α. If we consider the earlier example, then the α-connected components in figure 1 define the horizontal cuts for $\alpha \in \{0,1,2,4,5\}$. Note that the horizontal cut with $\alpha = 3$ is equiva-

• *Jeroen Lammers with RUG, E-mail: j.lammers@student.rug.nl.*

lent to the horizontal cut of $\alpha = 2$ due to there being no dissimilarities with value 3. During this paper we will refer to the process of taking a horizontal cut at level α of the α-tree as filtering the α-tree at level α.

There are some problems with using the L_p norm as a dissimilarity measure. The main problem is chaining. Chaining occurs when we have a gradient between two regions which should otherwise be separate. Due to the small differences in dissimilarity within the gradient the areas will be connected when they should not. An example of this is given in figure 3.

5	5	5	5
1	2	3	4
0	0	0	0
0	0	0	0

(a) Base image.

5	5	5	5
1	2	3	4
0	0	0	0
0	0	0	0

(b) 0-CC.

5	5	5	5
1	2	3	4
0	0	0	0
0	0	0	0

(c) 1-CC.

Fig. 3. An input image demonstrating chaining with its corresponding α-connected components using the absolute difference as a dissimilarity measure with 4-connectivity.

There have been multiple approaches on lessening the chaining effect. In this paper we will focus on the approach based on odd Gabor filters[21] which detects edges and changes the dissimilarity score based on the strength of the edge.

Another problem which often occurs is poor narrow object detection. This is due to the narrow object being easily broken up by noise. One of the proposed solutions is the use of even Gabor filters[20]. The even Gabor filters are used to detect the narrow objects and modify the dissimilarity score if narrow objects are detected.

2.2 Gabor filters

In this study we will be augmenting the dissimilarity value based on additional data. The additional edge [11, 21] and ridge [20] data will be computed using a bank of Gabor filters [14]. For the computation of the edge signal we will use the odd (imaginary) Gabor filter and for the ridge signal we will use the even (real) Gabor filter. These filters are defined as:

$$g_{odd}(x,y) = \exp\left(-\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}\right)\sin\left(2\pi\frac{x'}{\lambda}+\psi\right)$$
$$g_{even}(x,y) = \exp\left(-\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda}+\psi\right)$$

with $x' = x\cos\theta + y\sin\theta$, $y' = -x\sin\theta + y\cos\theta$, λ represents the wavelength, θ the orientation, ψ the phase offset, σ the standard deviation of the Gaussian factor and γ the spatial aspect ratio.

The bank will consist of Gabor filters in the edge connectivity directions, i.e. for 4-connectivity we will consider $\theta \in \{0, \frac{\pi}{2}\}$ and for 8-connectivity we will consider $\theta \in \{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\}$. Since we are interested in the edge signals and ridge signals in a specific direction, we will not use the phase offset and we will consider it to be zero in the rest of the study. In section 3 we will go into more depth on how we pick our values for λ, σ and γ.

Using this bank of Gabor filters we will compute the edge signal and the ridge signal in each channel of the image separately. The channels are then combined by taking the L^p-norm over the channels C of the image I.

$$e_\theta(x,y) = \left(\sum_{i=0}^{C} |(I_i * g_{odd})(x,y)|^p\right)^{1/p}$$
$$r_\theta(x,y) = \left(\sum_{i=0}^{C} |(I_i * g_{even})(x,y)|^p\right)^{1/p}$$

These signals will be affected by noise and texture in the original image. One way of solving this would be to threshold the obtained signals to remove the noise. However, we choose to use a logistic function instead due to it providing a less harsh cut-off of the signals compared to a threshold function.

$$sig(x) = \frac{1}{1+\exp\left(-a * \frac{x-b}{2}\right)} \tag{1}$$

where a denotes the slope of the transition and b the offset of the transition slope. Note that the output of the logistic function is in the range $(0,1)$. The resulting signals are defined as:

$$\begin{aligned}\hat{e}_\theta(x,y) &= sig(e_\theta(x,y))\\ \hat{r}_\theta(x,y) &= sig(r_\theta(x,y))\end{aligned} \tag{2}$$

We will have a weight corresponding to the edge signal (w_e) and a weight corresponding to the ridge signal (w_r). We will consider the strongest response and multiply the original dissimilarity by the corresponding weight. This weight is determined by linear interpolation between 1 and the edge/ridge weight based on the response of the logistic function.

$$\hat{\delta}(p,q) = \begin{cases} \delta(p,q)(1+((w_e-1)*\hat{e}_\theta(p))) & \text{If } \hat{e}_\theta \geq \hat{r}_\theta \\ \delta(p,q)(1+((w_r-1)*\hat{r}_\theta(p))) & \text{If } \hat{e}_\theta < \hat{r}_\theta \end{cases} \tag{3}$$

where θ is in the direction from p to q.

2.3 CDL filters

Let us consider the odd and even Gabor filters in a single dimension, as presented in figure 4 and 5, respectively. We can see that the odd Gabor filter roughly corresponds to the central difference around a pixel. We can also see that the even Gabor filter is similar to an inverted 1D Laplacian filter. Hence one might wonder if it is necessary to add the additional complexity of the Gabor filters. So rather than dealing with the added complexity of the Gabor filters we will consider the crude approximation of the Gabor filters given by the central difference of a pixel and its 1D Laplacian filter in order to detect edges and ridges in the desired directions.

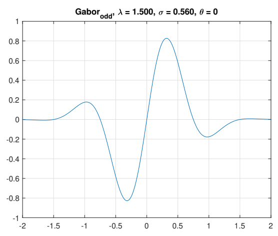


Fig. 4. Odd Gabor filter in 1D.

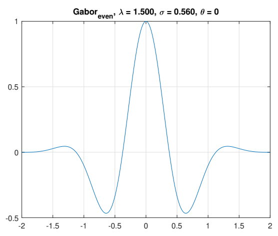


Fig. 5. Even Gabor filter in 1D

We will use a bank of central difference and 1D Laplacian filter to compute an approximation of the edge and ridge signal generated by the Gabor filters. The bank of filters will be based on the connectivity directions, similar to the Gabor filters. This means that in the case of 4-connectivity we will consider $\theta \in \{0, \frac{\pi}{2}\}$ and in the case of 8-connectivity we will consider $\theta \in \{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\}$.

$$\begin{array}{ccc}
\text{Central difference} & \text{1D Laplacian} & \theta \\
\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix} & 0 \\
\begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} & \begin{bmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & -1 \end{bmatrix} & \frac{\pi}{4} \\
\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} & \begin{bmatrix} 0 & -1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{bmatrix} & \frac{\pi}{2} \\
\begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & -1 \\ 0 & 2 & 0 \\ -1 & 0 & 0 \end{bmatrix} & \frac{3\pi}{4}
\end{array}$$

Using this bank of central difference (CD_θ) and 1D Laplacian (L_θ) filters we will compute the edge signal and ridge signal in each channel by convolving the input image with each kernel. We will combine the resulting channels by computing the L_p-norm over the channels, as we did for the Gabor filters.

$$e_\theta(x,y) = \left(\sum_{i=0}^{C} |(I_i * CD_\theta)(x,y)|^p \right)^{1/p}$$
$$r_\theta(x,y) = \left(\sum_{i=0}^{C} |(I_i * L_\theta)(x,y)|^p \right)^{1/p}$$

From this point we will apply the same post-processing on the signals as we did for the Gabor filters. We will transform the signals using a logistic function and add them to the original dissimilarity, as we did in equation 1, 2 and 3.

2.4 Difference filters

Lastly we will define the difference filters. The difference filters works differently compared to the previously mentioned detectors since it does not augment the normal dissimilarity score, but instead computes it in a different way. It is based on the forward difference (FD), backward difference (BD) and the central difference (CD) of a pixel in the directions of the connectivity and is computed for each channel separately. The signals of each channel are then combined using an L_p-norm. Each signal has a corresponding weight and they are combined as follows:

$$\delta(p) = |w_{FD} * FD(p) + w_{BD} * BD(p) + w_{CD} * CD(p)|$$

2.5 Fitness function

In order to train the model we will need a method of scoring how well our current model performs. For this we will be using the method discussed in [19]. The quality of a horizontal cut is defined using four scores. The area score (AS), the node index (NI), the depth index (DI) and the plateau length index (PLI).

2.5.1 Area Score

The area score is used to estimate how well a specific segmentation or ground truth is represented in the alpha tree. We will compute this using the over-merging error and the under-merging error defined below.

Assume that the ground truth has n segments denoted by G_i with $i \in \{1, 2, \ldots, n\}$. Now consider the image obtained after filtering the α-tree with a value $\hat{\alpha}$ and assume this image has m segments denoted by S_j with $j \in \{1, 2, \ldots, m\}$. Given a segment S_j, let us define the segment G_i which has the largest intersection with S_j as $\hat{G}_i$. Similarly, given a segment G_i, let $\hat{S}_j$ be the segment in the original image with the largest intersection with G_i. Let $|S_j \cap \hat{G}_i|$ and $|G_i \cap \hat{S}_j|$ be the size of the corresponding intersections. Using this we will define the under-merging error (UM) and the over-merging error (OM):

$$UM = \frac{1}{|I|} \sum_{j=1}^{m} \frac{(|\hat{G}_i| - |S_j \cap \hat{G}_i|)(|S_j \cap \hat{G}_i|)}{|\hat{G}_i|}$$
$$OM = \frac{1}{|I|} \sum_{i=1}^{n} \frac{(|\hat{S}_j| - |G_i \cap \hat{S}_j|)(|G_i \cap \hat{S}_j|)}{|\hat{S}_j|}$$

where $|I|$ denotes the area covered in the ground truth. Using the under-merging error and over-merging error we can compute the area score as follows:

$$AS = 1 - \sqrt{UM^2 + OM^2}$$

2.5.2 Node index

The node index represents the complexity of building the α-tree. This is represented by the number of nodes present in the tree. The number of nodes in an α-tree is bound by two times the number of pixels in the image. Thus the node index is defined as:

$$NI = 1 - \frac{|\mathcal{N}_\tau|}{2|I|}$$

where $|\mathcal{N}_\tau|$ is the number of nodes in the α-tree and $|I|$ the size of the image. Note that the node index is not dependent on a specific filtering of the α-tree. Also note that the node index is in the range $[0, 1]$ and is higher for α-trees with less nodes.

2.5.3 Depth index

The depth index represents the complexity in filtering the α-tree. The lower the desired alpha value, the less steps are necessary to perform a filter operation. The depth index for a given alpha value $\hat{\alpha}$ is defined as:

$$DI = 1 - \frac{\hat{\alpha}}{\alpha_\tau}$$

where α_τ is the largest alpha value in the α-tree. Note that the depth index is in the range $[0, 1]$ and is higher for low values of alpha.

2.5.4 Plateau length index

The plateau length index represents the robustness of the segmentation. We compute the area score over all alphas and consider the segment of alpha values around the optimal alpha value α_H where the area score is at least 90% of the optimal area score. The length of this segment is denoted as the plateau length (PL). This can be used to compute the plateau length index as follows:

$$PLI = \frac{PL}{\alpha_\tau}$$

where α_τ is the largest alpha value in the α-tree. Note that the plateau length index is in the range $[0, 1]$ and is higher if there is a larger interval of alpha values providing a good solution.

2.5.5 Computing the quality

In order to compute the quality score we will start by computing the area score for all alpha values. Then we will define α_H to be the alpha with the highest area score. We define the corresponding area score AS_H. Based on α_H we will compute the depth index, denoted by DI_H, and the plateau length index, denoted by PLI_H. Lastly we will compute the node index of the α-tree. The quality score can be computed as follows:

$$Q_H = w_1 AS_H + w_2 NI + w_3 DI_H + w_4 PLI_H$$

The choice of weights w_1, w_2, w_3 and w_4 depend on the properties desired.

3 Experimental setup

In this section we will provide an overview of the experimental setup we used, as well as some implementation-related optimisations and approximations used.

3.1 Defining a score for multiple ground truths

The fitness function introduced in section 2.5 provides a method of determining how well a single given ground truth is represented in an alpha tree. However, we want to train our model in such a way that it is able to segment the image on multiple levels, providing a good result for each. We need to define how we combine the individual scores of each ground truth to an overall score. We choose the geometric mean of the quality scores of the individual ground truths. Since the quality scores are in the range $[0,1]$ we will have that the geometric mean is only close to one if all scores are close to one. Conversely, if we have a single score not close to one, then it will drastically reduce the overall quality score. Hence we need a good segmentation of each of the ground truths in order to have a good overall quality score.

3.2 Fitness weights

For our experiments we chose to prioritise the area score component of the quality score by giving it a weight of 0.7. We chose this since this should result in better visual segmentations. For each of the other scoring components we picked a weight of 0.1.

3.3 Fitness function optimisation

In order to compute the optimal score we need to find the highest area score. A naive way of finding the optimal alpha would be to compute the area score for each alpha. However, if we consider a multi spectral image, the chance of having two dissimilarity scores which are exactly the same is very small, since even a small variation in one of the channels will result in a slightly different dissimilarity value. By design, the area score should not differ significantly for small changes in the dissimilarity. Thus, computing the area score for all values of alpha present in the α-tree would result in a lot of redundant computation. Therefore we will be computing a rougher estimation of the area score. For this we will consider m dissimilarity values. In order to make no assumption about the distribution of the alpha values, we will consider the array of sorted alpha values and let there be a total of $\#alpha$ alpha values. Then we can compute the indices of the elements considered in the sorted array as follows:

$$idx = \frac{(\#alpha - 1) * i}{m-1} \qquad i \in \{0,1,2,\ldots,m\}$$

For our experiments we considered $m = 400$. This was chosen experimentally in order for each run of the experiment to take about an hour on our AMD Opteron 6276 machine.

3.4 Testing and training data

We will consider a small data set of satellite images from the campus of the University of Groningen. The data set consists of four $1000x1000$ rgb images and three ground truth images of different levels of segmentation for each of them. These images can be found in figure 9, 10, 11 and 12 in the appendix. We will use each of the images to train the filters in order to find optimal parameters for each method. Note that each of the filters is trained on every single satellite image and its corresponding ground truths, so no batch learning is applied.

3.5 Parameter optimisation

In order to train our method in finding the optimal parameters for segmentation on the training data we will use Basian optimisation. We chose this method due to its efficiency for optimising black box functions with a relatively low number of evaluations. For our implementation we chose Metric Optimisation Engine[18] (MOE) due to its easy-to-use Python interface.

In order to apply MOE we need to bound the parameters that we want to optimise. Note that we want to increase the dissimilarity value in case we find an edge. However, we want to decrease the value in the ridge direction. Based on this we choose the parameters depicted in table 1, 2 and 3. Note that we fixed the p value of our L_p norms to 2, this was done because we encountered unexpected artefacts in the output if we did not use a multiple of 2.

Parameter	Min	Max
L_p-norm used as initial dissimilarity	2	2
λ parameter of the odd Gabor filter	0.5	5
σ parameter of the odd Gabor filter	0.5	2
γ parameter of the odd Gabor filter	0	2
L_p-norm used to reduce the edge signals	2	2
Slope of the sigmoid used for the edge signal	0.1	5
Center of the sigmoid used for the edge signal	0	40
Weight of the sigmoid used for the edge signal	1	10
λ parameter of the even Gabor filter	0.5	5
σ parameter of the even Gabor filter	0.5	2
γ parameter of the even Gabor filter	0	2
L_p-norm used to reduce the ridge signals	2	2
Slope of the sigmoid used for the ridge signal	0.1	5
Center of the sigmoid used for the ridge signal	0	40
Weight of the sigmoid used for the ridge signal	0	1

Table 1. Parameter bounds used in the Gabor filter optimisation

Parameter	Min	Max
L_p-norm used as initial dissimilarity	2	2
L_p-norm used to reduce the edge signals	2	2
Slope of the sigmoid used for the edge signal	0.1	5
Center of the sigmoid used for the edge signal	0	40
Weight of the sigmoid used for the edge signal	1	10
L_p-norm used to reduce the ridge signals	2	2
Slope of the sigmoid used for the ridge signal	0.1	5
Center of the sigmoid used for the ridge signal	0	40
Weight of the sigmoid used for the ridge signal	0	1

Table 2. Parameter bounds used in the CDL filter optimisation

Parameter	Min	Max
L_p-norm used to reduce the FD signals	2	2
Weight of the FD signal	-1	1
L_p-norm used to reduce the BD signals	2	2
Weight of the BD signal	-1	1
L_p-norm used to reduce the CD signals	2	2
Weight of the CD signal	-1	1

Table 3. Parameter bounds used in the difference filter optimisation

3.6 Experimental setup

We have implemented the α-tree algorithm and scoring procedure in C++[10]. This program is called through a Python script which oversees the generation of new parameters via MOE. We run multiple instances of the program in order to attain pseudo parallelism, i.e. when a process is finished, we update our known results to obtain new parameters. We do not wait for all processes to finish. In our experiments we run four processes for each of the four satellite images and each of the three methods. During the execution we store the parameters used, the final score, the scores for each ground truth and the optimal alpha value of each ground truth corresponding to that score. Note that we only consider 4-connectivity in our experiments. We ran the training code for 3 days.

3.7 Processing experiment output

For our experiments we obtain data for each set of parameters considered. We will only consider the best performing sets of parameters of each satellite image and method combination for further analysis.

These parameters will be tested on the other satellite images in order to observe the robustness of the filter for other similar input data.

4 Results and discussion

In this section we will present the results obtained for the experiments. We only consider the overall quality score of each filter, which is based on the combination of the three quality scores corresponding to the ground truth images.

4.1 Optimal filters

Table 4 contains the best 4 scores obtained from the experiments for each combination of method and satellite image. Table 9, 10 and 11 contain the parameters of the best performing runs.

In table 4 we can observe that the Gabor method scores consistently higher than its simplified counterpart CDL and the difference based method. However, we can also see that the difference in the scores is only a few percent. We can also observe that the top 4 results have very similar score values. This would imply that we have encountered a plateau while training. Due to the non-linearity of the parameter space this does not have to imply that we found the optimal solution. It might be possible to find a better solution if we run the optimisation for a longer period of time.

Image	Gabor	CDL	Difference
1	0.884655	0.878461	0.863846
	0.884124	0.877768	0.863846
	0.882817	0.876612	0.863846
	0.880114	0.874687	0.863846
2	0.882537	0.874265	0.867338
	0.881505	0.87406	0.867338
	0.881462	0.87405	0.867338
	0.880935	0.873971	0.867338
3	0.918579	0.910538	0.88335
	0.918332	0.910059	0.88335
	0.917029	0.908741	0.88335
	0.913546	0.90849	0.88335
4	0.849467	0.849291	0.831299
	0.849245	0.847714	0.831299
	0.849187	0.847617	0.831299
	0.849054	0.84654	0.831299

Table 4. Top 4 best scores observed for each method on each image.

4.2 Robustness

Tables 5, 6, 7 and 8 show the application of the highest scoring filter for a given image on the other images. The filter considered is marked in bold.

Image	Gabor	CDL	Difference
1	**0.884655**	**0.878461**	**0.863846**
2	0.869684	0.879297	0.867338
3	0.88327	0.915907	0.874438
4	0.84334	0.84704	0.831299

Table 5. The resulting scores of using the optimal filter for image 1 on the other images.

Image	Gabor	CDL	Difference
1	0.883873	0.874295	0.863846
2	**0.882537**	**0.874265**	**0.867338**
3	0.882946	0.912175	0.874438
4	0.850335	0.844834	0.831299

Table 6. The resulting scores of using the optimal filter for image 2 on the other images.

Image	Gabor	CDL	Difference
1	0.884095	0.8755	0.841331
2	0.88162	0.876179	0.852374
3	**0.918579**	**0.910538**	**0.88335**
4	0.848634	0.841887	0.827155

Table 7. The resulting scores of using the optimal filter for image 3 on the other images.

Image	Gabor	CDL	Difference
1	0.875769	0.875633	0.863846
3	0.883069	0.880293	0.867338
3	0.919213	0.914186	0.874438
4	**0.849467**	**0.849291**	**0.831299**

Table 8. The resulting scores of using the optimal filter for image 4 on the other images.

As we can see in tables 5-8, in almost all cases the score on image 3 is higher than on the image it was trained on. Similarly, in all cases the scores for image 4 are the lowest. We can also observe that the Gabor score is consistently higher than the CDL score, if only by a small amount. Similarly, both Gabor and CDL score consistently higher than the difference method. All parameters perform within a few percent margin of the parameters for a given image which would indicate that the parameters define robust filters.

4.3 Visual results

This section will discuss the visual results from figures 13-24 in the appendix. These figures contain the ground truths and the segmentations corresponding to the highest scoring parameters according to the experiment.

We will start by noting that we expect the alpha values to be increasing for decreasing level of detail in the ground truths. However, if we look at the optimal alpha values found for the Gabor method for the second image we can see that the alpha value decreases between the first and the second ground truth. This is also the case for the the first and the second image when considering the difference based method. Due to the nature of the ground truths this should not happen, so this would imply that there might be a problem with the score computation.

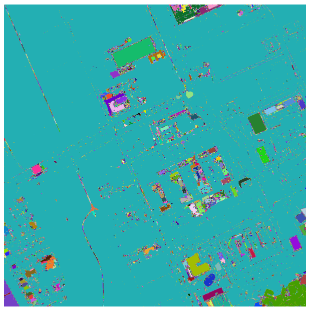
Next we will note that there are 2 cases in which we observe the same alpha value for different ground truths. More specifically, the solution of image 1 for ground truth 2 and 3 for the Gabor method, image 1 with ground truth 2 and 3 for the difference method and image 2 with ground truth 1 and 2 for the difference method. The lack of different alpha values indicates that the approximation of the area score was not fine enough.

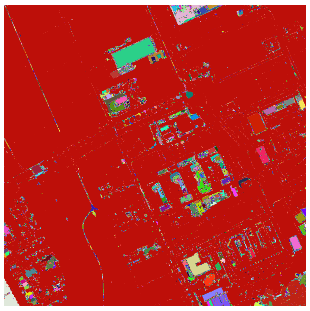
Lastly, most of the results look to be over-merged, i.e. we have that most of the image consist of one very large region. An example of this is shown in figure 6. We expect this segmentation to be present for larger alpha values (higher in the tree). Thus, it seems that the optimisation method consistently cuts the tree too high.

If we consider the first non-zero alpha value used by the area score approximation then we obtain the result in figure 7. As we can see, the resulting segmentation is visually significantly better than the one proposed by the experiment for both the Gabor method and CDL method. For the difference method we observe under-merging. This would indicate that there is a better solution present between these alpha values. Thus, we have that either the area score is not a good indicator for visual similarity or that there is a problem with its implementation for our experiment.

If we were to hand pick alpha values, we can obtain the segmentation from figure 8. We picked a smaller alpha value for the alpha Gabor method in order to find a solution with more detail. For the difference method we pick an alpha value between the under-merged result and the over-merged result. For the Gabor method we can see that we obtain more segmentations in regions such as roads. Hence one could argue that it provides a more detailed segmentation and that

(a) Ground truth 1 of Image 1.


(b) Gabor $\alpha = 0.326793$.


(c) CDL $\alpha = 3.000000$.

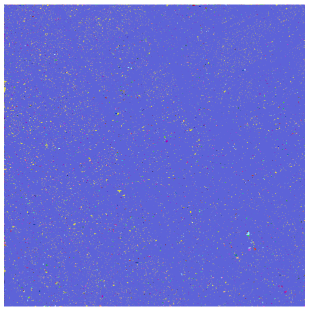
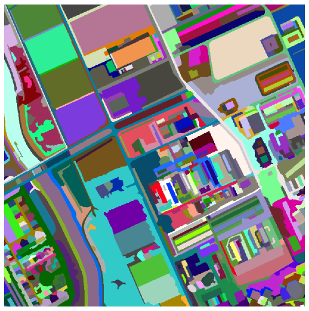
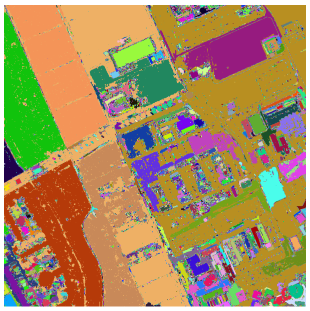
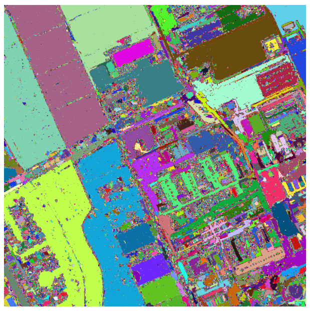

(d) Difference $\alpha = 0.548409$.

Fig. 6. Example of over-merging for the optimal segmentation of image 1 corresponding to the first ground truth.


(a) Ground truth 1 of Image 1.


(b) Gabor $\alpha = 0.128882$.


(c) CDL $\alpha = 0.554141$.

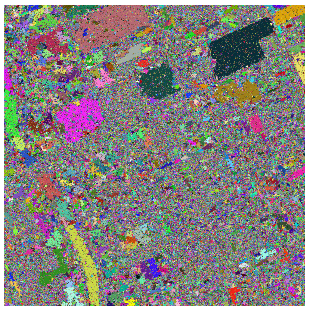
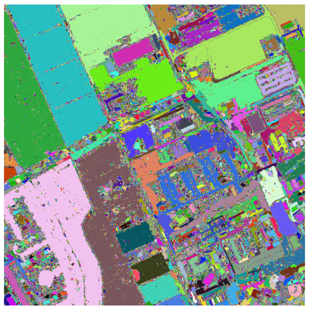

(d) Difference $\alpha = 8.9987e-5$.

Fig. 7. First non-zero alpha value considered by the area function approximation.


(a) Gabor $\alpha = 0.03$.

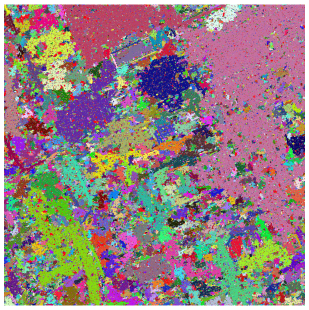

(b) Difference $\alpha = 0.09$.

Fig. 8. Example of better segmentation being present in the α-tree.

our approach in reducing the number of alpha values considered in the area score is not optimal. For the difference method we can see that the result has a lot of noise and irregular regions compared to the results of the Gabor and CDL results. Even though the scores are very similar, the resulting segmentation is visually worse.

5 Conclusion

From the observations made in section 4 we can conclude that even though the theoretical scores are similar for all the methods, the visual results of the difference method are worse than those of the other methods. We have also seen that the parameters obtained from training on a single image can be used on the other images in the data set in order to obtain similar scores as a result. We can also conclude that our approximation of the area score is too coarse which results in less than ideal segmentations. This could also be caused due to a problem in the area score computation, since we found other results which provided a better visual result but did not produce the highest area score. This would imply that either the area score is not a sufficient measure to determine the dissimilarity of the segmentation and the ground truth or that we made an error in its implementation. Determining which of these is the case could be determined in further research. We can conclude that the results of the Gabor method and the CDL method are very similar, both in score and visual results. Since the CDL method only has 6 degrees of freedom in the choice of parameters compared to the 12 degrees of freedom for the Gabor case, it might be an interesting alternative due to the lower complexity. From our results we observe that even though the difference method provides similar scores, it does not provide a good visual segmentation. This could indicate that the method itself is flawed or that the parameter optimisation was insufficient. Due to the limited time we were able to run the experiments and the aforementioned problems with the area score, we suspect that the parameter optimisation was insufficient. Hence the method might need further testing to determine its feasibility.

6 Future research

In this paper we have mainly looked at augmenting the α-tree algorithm with data from an edge detector and a ridge detector. However, there are also other extensions made to alpha trees to improve their segmentation performance, namely constrained connectivity α-ω-trees [16] and contrast based α-trees [17]. These methods could be combined with the methods discussed in this paper to possibly provide better results.

In this paper we have proposed a method of combining the edge signal and ridge signal with the dissimilarity score using a naive approach. However, it might be interesting to look at different ways of combining the edge signal and ridge signal of each channel. One method would be to have one vote per channel for either edge signal or ridge signal and only consider the highest voted option for the manipulation of the dissimilarity. In our approach we used both the edge signal and the ridge signal to change the dissimilarity, however one might be able to conceive of a better approach.

Another related topic would be to consider the work done using morphological (attribute) profiles [4, 13] on high resolution satellite images for segmentation purposes, especially since it is currently in use in the Orfeo ToolBox [8] which is actively used in the QGIS application [15]. We did not consider morphological profiles in this paper due to them being not well defined for multi spectral images. However, it might be interesting to compare these methods to the current α-tree based methods which do work on multi spectral images.

Lastly, we propose to consider the Tsetlin machine [6]. It is a simple automata that can be used to learn patterns in an efficient way. Originally it was mainly used on natural language problems, but it has since been extended to convolution problems [7]. Since its original publication there have been multiple extensions adding continuous input [3], continuous output [1, 5] and better efficiency [2]. Due to the simple architecture it is easily implemented directly in hardware and hence it might be interesting to see if it can be extended to be used in finding optimal dissimilarity values for the α-tree construction.

References

[1] D. Abeyrathna, O.-C. Granmo, and M. Goodwin. Convolutional regression tsetlin machine: An interpretable approach to convolutional regression. In *2021 6th International Conference on Machine Learning Technologies*, pages 65–73, 2021.

[2] K. D. Abeyrathna, O.-C. Granmo, R. Shafik, A. Yakovlev, A. Wheeldon, J. Lei, and M. Goodwin. A novel multi-step finite-state automaton for arbitrarily deterministic tsetlin machine learning. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 108–122. Springer, 2020.

[3] K. D. Abeyrathna, O.-C. Granmo, X. Zhang, and M. Goodwin. A scheme for continuous input to the tsetlin machine with applications to forecasting disease outbreaks. In *International conference on industrial, engineering and other applications of applied intelligent systems*, pages 564–578. Springer, 2019.

[4] M. Dalla Mura, J. A. Benediktsson, B. Waske, and L. Bruzzone. Morphological attribute profiles for the analysis of very high resolution images. *IEEE Transactions on Geoscience and Remote Sensing*, 48(10):3747–3762, 2010.

[5] K. Darshana Abeyrathna, O.-C. Granmo, X. Zhang, L. Jiao, and M. Goodwin. The regression tsetlin machine: a novel approach to interpretable nonlinear regression. *Philosophical Transactions of the Royal Society A*, 378(2164):20190165, 2020.

[6] O.-C. Granmo. The tsetlin machine–a game theoretic bandit driven approach to optimal pattern recognition with propositional logic. *arXiv preprint arXiv:1804.01508*, 2018.

[7] O.-C. Granmo, S. Glimsdal, L. Jiao, M. Goodwin, C. W. Omlin, and G. T. Berge. The convolutional tsetlin machine. *arXiv preprint arXiv:1905.09688*, 2019.

[8] M. Grizonnet, J. Michel, V. Poughon, J. Inglada, M. Savinaud, and R. Cresson. Orfeo toolbox: Open source processing of remote sensing images. *Open Geospatial Data, Software and Standards*, 2(1):1–8, 2017.

[9] L. Gueguen, G. Ouzounis, M. Pesaresi, and P. Soille. Tree based representations for fast information mining from vhr images. In *Proceedings of the ESA-EUSCJRC Eight Conference on Image Information Mining, Prof. Mihai Datcu, Ed*, 2012.

[10] J. Lammers. Source code of the experiments. https://github.com/JeroenLam/Research-Internship-Finding-an-optimal-dissimilarity-measure-for-hierarchical-segmentation.

[11] R. Mehrotra, K. R. Namuduri, and N. Ranganathan. Gabor filter-based edge detection. *Pattern recognition*, 25(12):1479–1494, 1992.

[12] G. K. Ouzounis and P. Soille. The alpha-tree algorithm. *JRC Scientific and Policy Report*, 2012.

[13] M. Pesaresi and J. A. Benediktsson. A new approach for the morphological segmentation of high-resolution satellite imagery. *IEEE transactions on Geoscience and Remote Sensing*, 39(2):309–320, 2001.

[14] N. Petkov. Biologically motivated computationally intensive approaches to image pattern recognition. *Future Generation Computer Systems*, 11(4-5):451–465, 1995.

[15] QGIS Development Team. *QGIS Geographic Information System*. QGIS Association,

[16] P. Soille. Constrained connectivity and connected filters. *IEEE Trans. Pattern Anal. Mach. Intell*, 30(7):1132–1145, 2008.

[17] P. Soille. Preventing chaining through transitions while favouring it within homogeneous regions. In *International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing*, pages 96–107. Springer, 2011.

[18] Yelp. Metric optimization engine. https://github.com/Yelp/MOE. Accessed: 2022-09-01.

[19] X. Zhang and M. H. Wilkinson. Alpha-tree evaluation; part one: Horizontal cut.

[20] X. Zhang and M. H. Wilkinson. Improving detection of narrow objects in alpha trees.

[21] X. Zhang and M. H. Wilkinson. Preventing chaining in alpha-trees using gabor filters. In *International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing*, pages 268–280. Springer, 2019.

7 Appendix

Parameter	Image 1	Image 2	Image 3	Image 4
L_p-norm used as initial dissimilarity	2	2	2	2
λ parameter of the odd Gabor filter	0.687754222739	3.30613630108	4.80444215263	0.53802506389
σ parameter of the odd Gabor filter	1.80754480804	1.9793400773	1.78400777427	1.41588492388
γ parameter of the odd Gabor filter	1.90965542569	0.16063958295	1.45830147376	1.92489663945
L_p-norm used to reduce the edge signals	2	2	2	2
Slope of the sigmoid used for the edge signal	4.71057580966	2.76742570862	0.980770298863	1.83707721632
Center of the sigmoid used for the edge signal	2.20726967437	35.447594255	31.6829283623	33.6939567704
Weight of the sigmoid used for the edge signal	9.64882600945	9.34930788421	7.55882350425	8.86248470095
λ parameter of the even Gabor filter	4.75626828672	1.11274977766	0.609726177153	4.51366488161
σ parameter of the even Gabor filter	1.67120103542	1.72843450503	0.532210464513	0.784892135836
γ parameter of the even Gabor filter	0.271139200934	1.52546115205	0.133567091554	0.733033050066
L_p-norm used to reduce the ridge signals	2	2	2	2
Slope of the sigmoid used for the ridge signal	4.86442763908	0.792703702507	1.19131510712	4.54046846262
Center of the sigmoid used for the ridge signal	31.9079901748	32.8320823182	22.4187424193	24.9752319999
Weight of the sigmoid used for the ridge signal	0.12888197051	0.0475527444329	0.432198771971	0.0198284052254

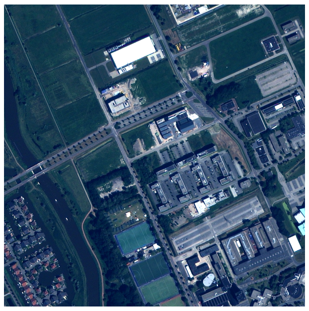
Table 9. Best parameters found for the Gabor method after training.

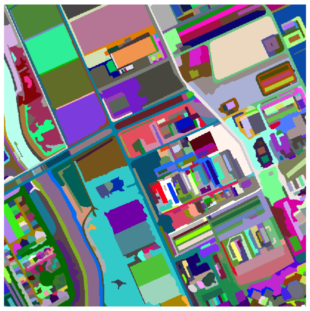
Parameter	Image 1	Image 2	Image 3	Image 4
L_p-norm used as initial dissimilarity	2	2	2	2
L_p-norm used to reduce the edge signals	2	2	2	2
Slope of the sigmoid used for the edge signal	4.1311660112	4.61787562555	3.90094927812	4.20617431257
Center of the sigmoid used for the edge signal	39.2767050024	39.5710541573	39.5204182578	32.0911939161
Weight of the sigmoid used for the edge signal	9.95003579969	4.73134338354	2.21786765531	9.73334079885
L_p-norm used to reduce the ridge signals	2	2	2	2
Slope of the sigmoid used for the ridge signal	3.59072939661	4.25367866441	4.91274136593	4.8571271544
Center of the sigmoid used for the ridge signal	39.8789519468	36.8696638155	0.0300276284882	38.6294029714
Weight of the sigmoid used for the ridge signal	0.554141000734	0.57784612174	0.0659786532633	0.535444049565

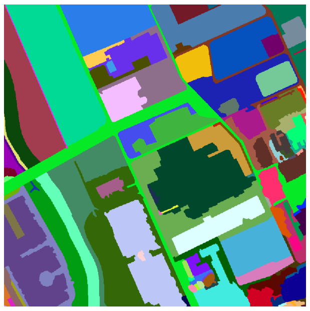
Table 10. Best parameters found for the CDL method after training.

Parameter	Image 1	Image 2	Image 3	Image 4
L_p-norm used to reduce the FD signals	2	2	2	2
Weight of the FD signal	1	1	-1	-1
L_p-norm used to reduce the BD signals	2	2	2	2
Weight of the BD signal	1	1	1	-1
L_p-norm used to reduce the CD signals	2	2	2	2
Weight of the CD signal	-1	-1	-1	1

Table 11. Best parameters found for the difference method after training.


(a) Image 1.


(b) Fine ground truth image 1.


(c) Medium ground truth image 1.

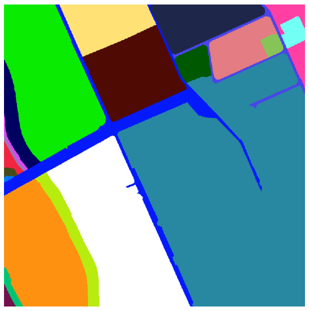
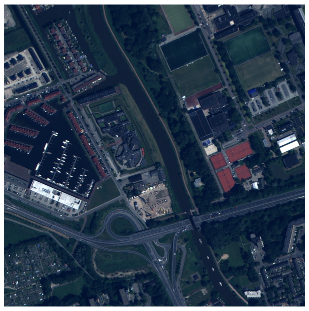
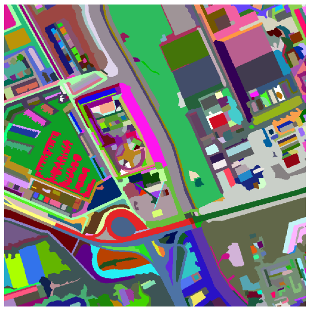
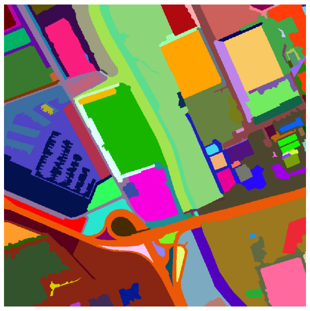

(d) Coarse ground truth image 1.

Fig. 9. Image 1 and its corresponding ground truths of different segmentation levels.


(a) Image 2.


(b) Fine ground truth image 2.


(c) Medium ground truth image 2.

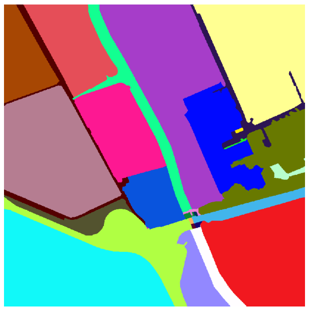
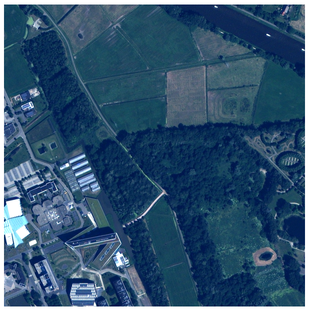
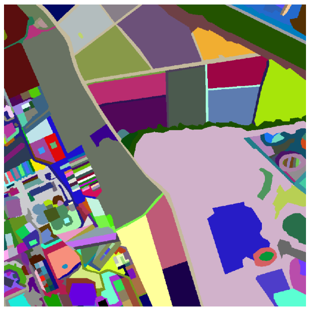
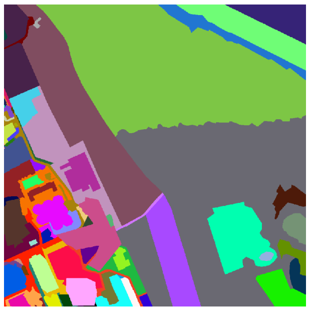

(d) Coarse ground truth image 2.

Fig. 10. Image 2 and its corresponding ground truths of different segmentation levels.


(a) Image 3.


(b) Fine ground truth image 3.


(c) Medium ground truth image 3.

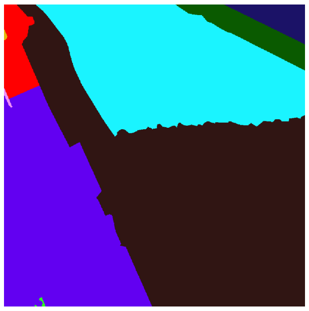
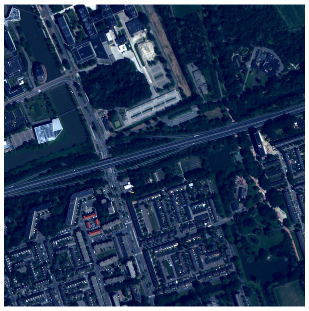
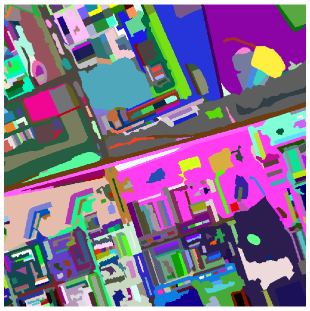
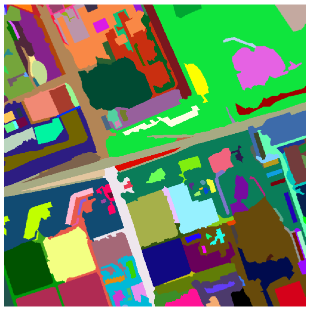

(d) Coarse ground truth image 3.

Fig. 11. Image 3 and its corresponding ground truths of different segmentation levels.


(a) Image 4.


(b) Fine ground truth image 4.


(c) Medium ground truth image 4.

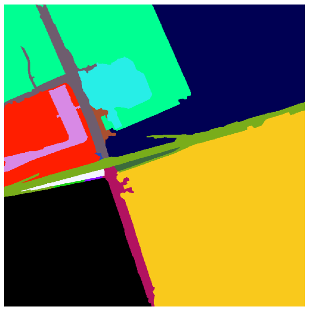
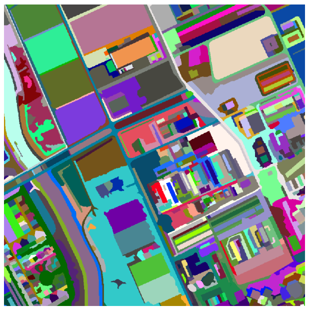
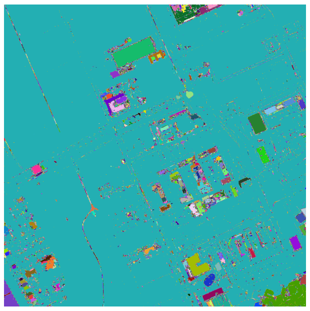
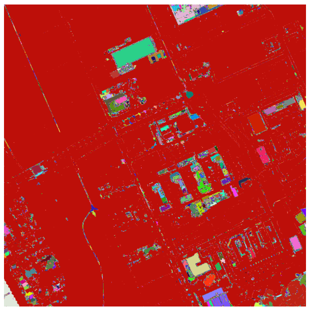

(d) Coarse ground truth image 4.

Fig. 12. Image 4 and its corresponding ground truths of different segmentation levels.


(a) Ground truth 1 of Image 1.


(b) Gabor $\alpha = 0.326793$.


(c) CDL $\alpha = 3.000000$.

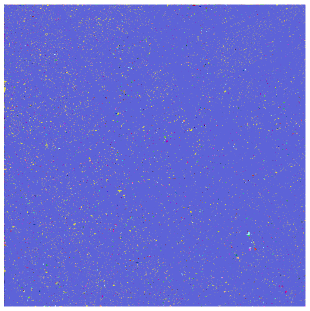
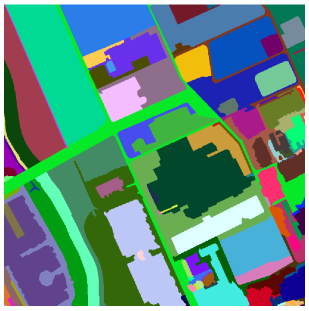
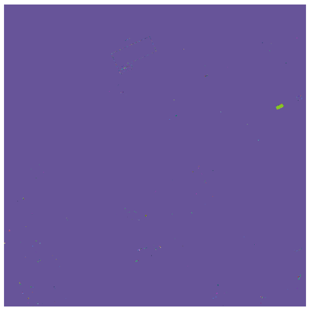
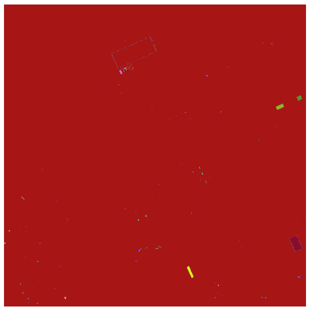

(d) Difference $\alpha = 0.548409$.

Fig. 13. Ground truth 1 of Image 1 and the corresponding segmentation using the best scoring filters.


(a) Ground truth 2 of Image 1.


(b) Gabor $\alpha = 2.577640$.


(c) CDL $\alpha = 37.00020$.

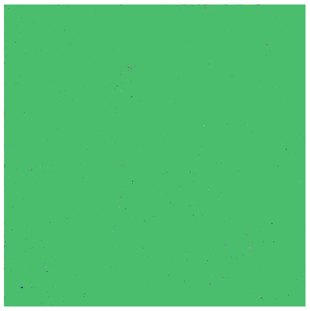
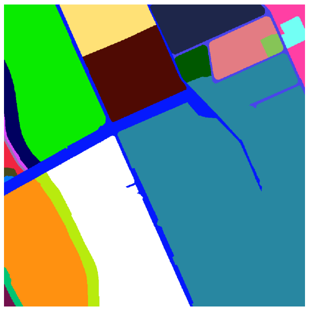
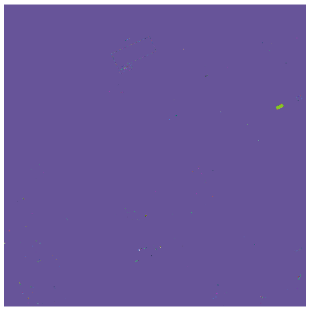
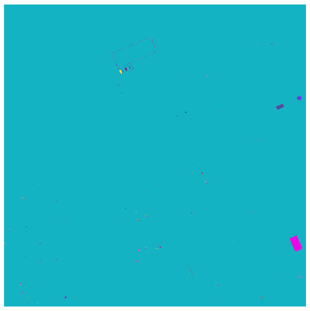

(d) Difference $\alpha = 2.362970$.

Fig. 14. Ground truth 2 of Image 1 and the corresponding segmentation using the best scoring filters.


(a) Ground truth 3 of Image 1.


(b) Gabor $\alpha = 2.577640$.


(c) CDL $\alpha = 39.800100$.

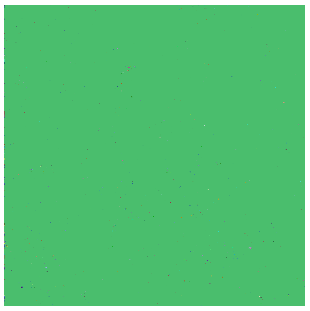
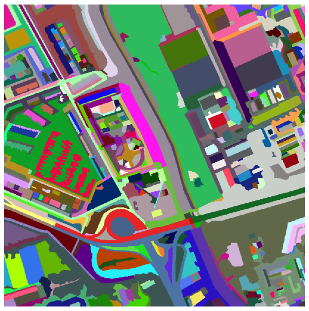
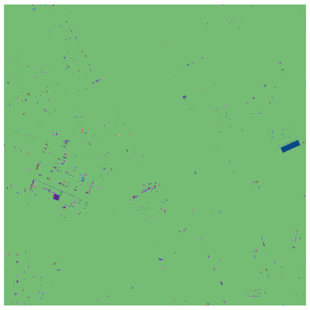
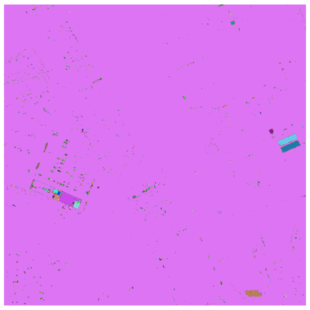

(d) Difference $\alpha = 2.362970$.

Fig. 15. Ground truth 3 of Image 1 and the corresponding segmentation using the best scoring filters.

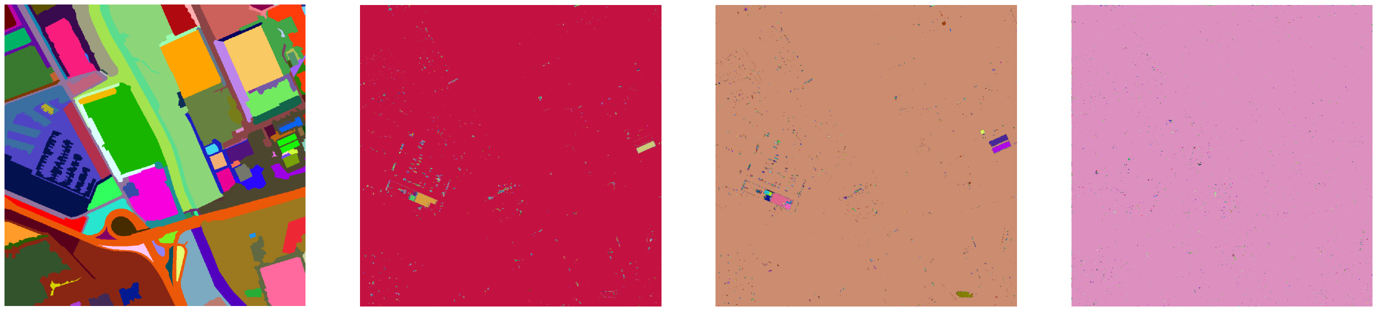

(a) Ground truth 1 of Image 2.


(b) Gabor $\alpha = 0.332869$.


(c) CDL $\alpha = 5.000000$.

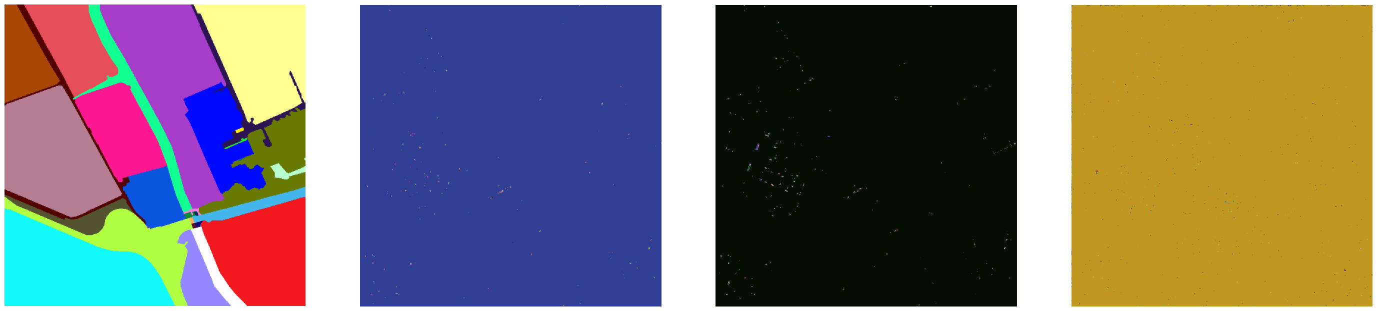
(d) Difference $\alpha = 0.754669$.

Fig. 16. Ground truth 1 of Image 2 and the corresponding segmentation using the best scoring filters.



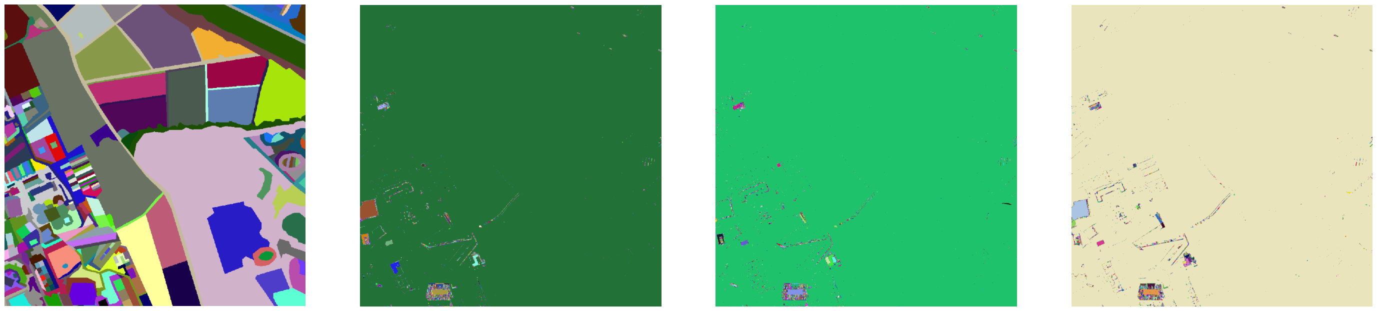
(a) Ground truth 2 of Image 2. (b) Gabor $\alpha = 0.237764$. (c) CDL $\alpha = 5.000020$. (d) Difference $\alpha = 0.754669$.

Fig. 17. Ground truth 2 of Image 2 and the corresponding segmentation using the best scoring filters.



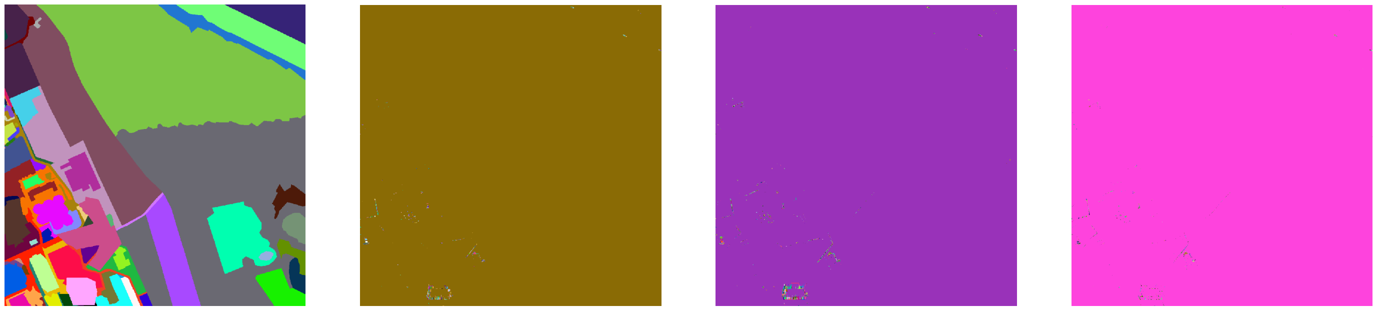
(a) Ground truth 3 of Image 2. (b) Gabor $\alpha = 9.349310$. (c) CDL $\alpha = 15.000000$. (d) Difference $\alpha = 1.676260$.

Fig. 18. Ground truth 3 of Image 2 and the corresponding segmentation using the best scoring filters.



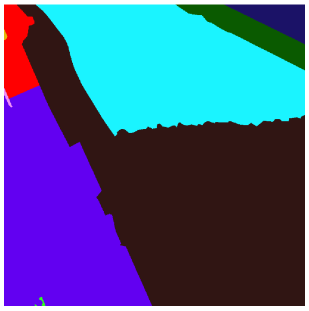
(a) Ground truth 1 of Image 3. (b) Gabor $\alpha = 2.593190$. (c) CDL $\alpha = 0.461851$. (d) Difference $\alpha = 13.844400$.

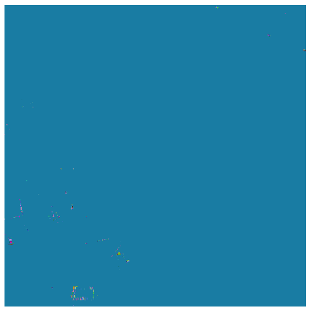
Fig. 19. Ground truth 1 of Image 3 and the corresponding segmentation using the best scoring filters.

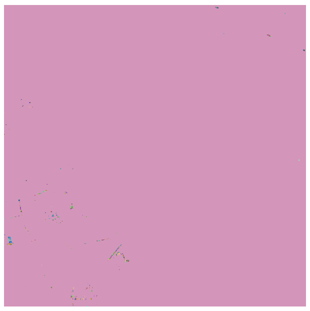


(a) Ground truth 2 of Image 3. (b) Gabor $\alpha = 7.558820$. (c) CDL $\alpha = 0.923701$. (d) Difference $\alpha = 34.725000$.

Fig. 20. Ground truth 2 of Image 3 and the corresponding segmentation using the best scoring filters.


(a) Ground truth 3 of Image 3.


(b) Gabor $\alpha = 8.211780$.


(c) CDL $\alpha = 1.163190$.

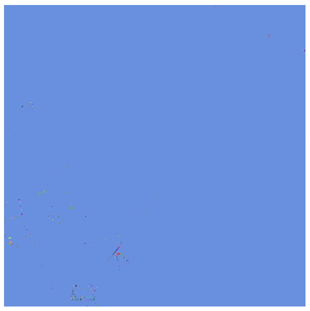
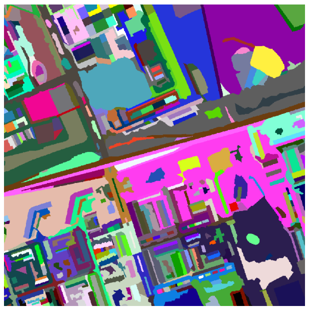
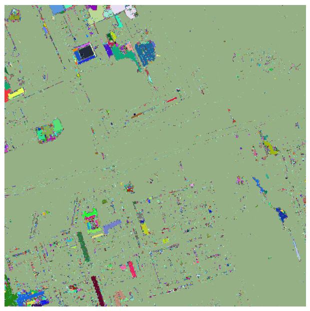
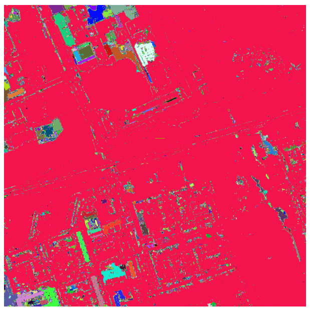

(d) Difference $\alpha = 32.020900$.

Fig. 21. Ground truth 3 of Image 3 and the corresponding segmentation using the best scoring filters.


(a) Ground truth 1 of Image 4.


(b) Gabor $\alpha = 0.039657$.


(c) CDL $\alpha = 2.000000$.

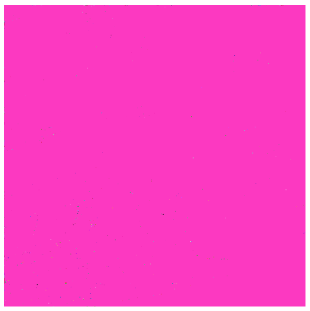
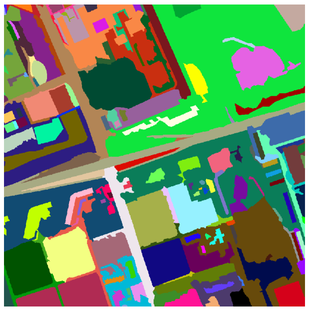
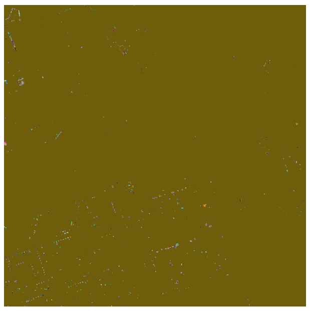

(d) Difference $\alpha = 2.415470$.

Fig. 22. Ground truth 1 of Image 4 and the corresponding segmentation using the best scoring filters.


(a) Ground truth 2 of Image 4.

(b) Gabor $\alpha = 0.158627$.


(c) CDL $\alpha = 9.000000$.

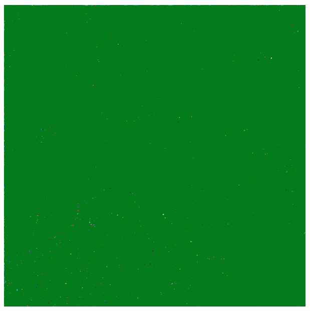
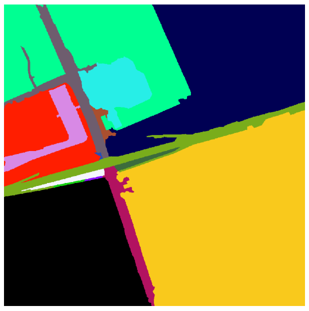
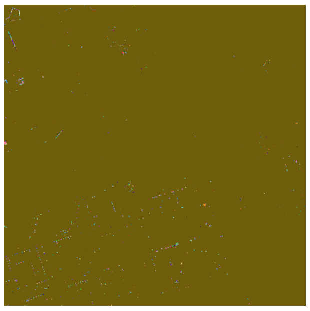

(d) Difference $\alpha = 1.553070$.

Fig. 23. Ground truth 2 of Image 4 and the corresponding segmentation using the best scoring filters.


(a) Ground truth 3 of Image 4.

(b) Gabor $\alpha = 345.637000$.


(c) CDL $\alpha = 399.067000$.

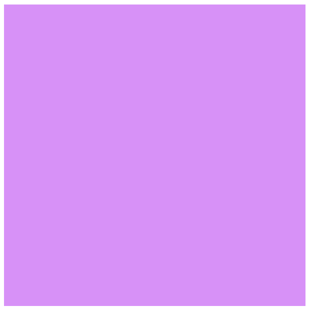

(d) Difference $\alpha = 16.492400$.

Fig. 24. Ground truth 3 of Image 4 and the corresponding segmentation using the best scoring filters.