

# Erstellung eines aktuellen wolkenfreien Satellitenbild-Komposit mit der Google Earth Engine am Beispiel von Simbabwe

Janusch Jehle

*Msc. Umweltwissenschaften, Albert-Ludwigs-Universität Freiburg*

---

## Abstract

Die Erstellung eines wolkenfreien Satellitenbild-Komposit ist in der Fernerkundung die Voraussetzung für jede weitere spektrale Analyse im panchromatischen Spektrum. In der vorliegenden Arbeit werden mithilfe der Google Earth Engine zwei Methoden zur Erstellung eines aktuellen wolkenfreien Satellitenbild-Komposit am Beispiel von Simbabwe vorgestellt und verglichen. Zum einen die Perzentil-Methode und zum anderen eine Erweiterung der Greenest-Pixel-Methode. Die Perzentil-Methode hat sich als äußerst robust für Gebiete mit schwacher bis mittlerer Wolkendichte erwiesen, die Greenest-Pixel-Methode ist durch das Detektieren und maskieren von Wolken flexibler auch für Gebiete mit hoher Wolkendichte einzusetzen, produziert vereinzelt jedoch noch Artefakte und muss weiter verbessert werden

---



In der Einführung soll zu Beginn die Notwendigkeit und Bedeutung eines Wolkenfreien Satellitenbild-Komposit für die Auswertung von Fernerkundungsdaten erläutert werden. Es folgt ein kurz Vorstellung der Google Earth Engine und einigen Anwendungsbeispielen. Im Methodenteil wird erst auf die verwendeten Satellitendaten eingegangen und im Anschluss die beiden Methoden vorgestellt. Nach einer kurzen Präsentation der Ergebnisse werden Stärken und Schwächen sowie mögliche Verbesserungen der beiden Methoden diskutiert.

## 1. Einführung

Über 70 Prozent der Erdoberfläche ist zu jeder Zeit von Wolken bedeckt[1], besonders in den Tropen ist das für die Optische Fernerkundung ein großes Problem [2]. Satellitensensoren, die im sichtbaren Lichtspektrum arbeiten, werden konstant durch Wolken am Blick auf die Erdoberfläche gehindert, oder durch Wolkenschatten gestört. Die Wolken oder Schatten der Wolken können dabei die Nutzung und Auswertung der aufgenommenen Satellitenbilder nicht nur erschweren sondern unmöglich machen, dass erstellen von wolkenfreien Satellitenbild-Kompositen ist damit von zentraler Bedeutung für die Fernerkundung[1].

### 1.1. Google Earth Engine

Die Google Earth Engine (GEE) ist eine kostenlose webbasierte Cloud-Plattform mit eigener API zum prozessieren von GIS Daten und einem Datenkatalog an frei verfügbaren Satellitendaten von 11 PB, darunter die Daten der NASA Programme Landsat und MODIS, sowie die Sentinel Missionen der ESA und viele weitere [3].

Neben dem Datenkatalog bietet GEE viele sehr leistungsstarke Server mit denen Berechnungen völlig neuer Größenordnungen möglich sind. Beispielhaft dafür ist die Studie von Hansen et al. [4] zur weltweiten Waldbestandsänderung, in der ein Gebiet von 128 Millionen  $km^2$  mit einer 30 Meter Auflösung untersucht wurde, was 143 Billionen Pixeln von Landsat Daten entspricht. Diese Berechnungen, die bei einem einzigen PC 16 Jahre benötigt hätten, waren über die GEE in weniger als 2 Tagen möglich. Weiter Beispiele sind die Studie von Pekel et al. [5] zur weltweiten Änderung von Oberflächenwasser und die webbasierte Dienste wie die Map Of Life, Global Forest Watch, Malaria Risk Mapping und Collect Earth [6].

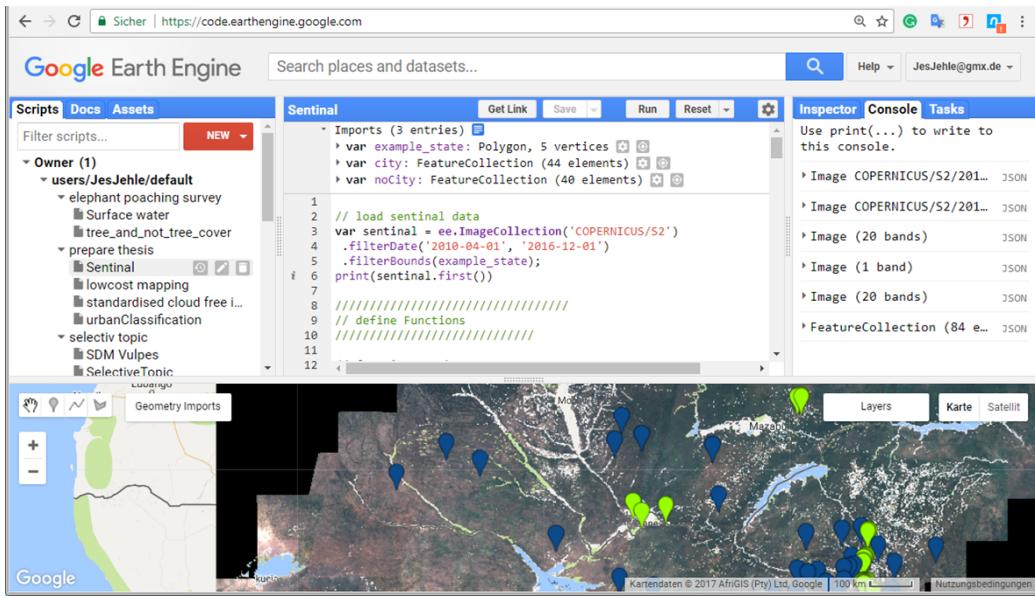


Abbildung 1: GEE Interface mit Arbeitsbereich oben und Kartenbereich unten

Das GEE Interface, zu sehen in Abbildung 1, besteht aus einem Arbeitsbereich (oben) und einer interaktiven Karte (unten), der Arbeitsbereich ist aufgeteilt in einen Code Editor (mitte), einem Bedienfeld in dem ein Überblick über Datensets, Algorithmen und Skripte gegeben wird (links) und eins, in dem die Konsole und detaillierte Informationen zur Karte und Daten zu finden sind (rechts). Alle Algorithmen und Objekte der GEE sind gut dokumentiert und es gibt eine Vielzahl an Beispielskripten und Tutorials. Programmiert wird in JavaScript oder Python, für Python gibt es jedoch kaum Dokumentation weshalb in der vorliegenden Arbeit mit JavaScript gearbeitet wird.

## 2. Methoden

### 2.1. Daten

Zur Herstellung eines aktuellen wolkenfreien Satellitenbild-Komposit für Botswana werden 1810 multispektrale Sentinel 2 Satellitenbilder im Top-Of-Atmosphere Reflectance (TOA Level-1C) Format verwendet welche eine Fläche von  $582.000 \text{ km}^2$  abdecken und im Zeitraum von 01-01-2016 bis 30-12-2016 aufgenommen wurden. Die Sentinel 2 Satelliten bewegen sich auf einer

sonnensynchronen - polaren Umlaufbahn. Sie folgen dabei einer festen Route, die durch das WRS-2 (Worldwide Reference System Katalog für Satelliten) definiert ist. Sentinel 2 besteht aus 2 Satelliten mit phasenverschobenen Umlaufbahnen, die gemeinsam in einem Rhythmus von 6 Tagen die gesamte Landfläche zwischen  $84^{\circ}N$  und  $56^{\circ}S$  in Kacheln (Tiles) abdecken. Jeder Satellit besitzt ein Multi-Spectral-Instrument (MSI) das 12 Bände im panchromatischen, nahen-Infrarot und Kurzwellen-Spektrum des Infrarot produziert. Im panchromatischen Spektrum besitzen die Sensoren eine Auflösung von 10 Meter pro Pixel [7]. Die TOA Level-1C Satellitenbilder decken jeweils Tiles von  $110 \times 110$  Km ab und besitzen eine Größe von 500 MB [7].

Die Sentinel Daten welche in GEE zu Verfügung stehen werden als Level-1C Produkte bezeichnet. Die Level reichen von 0 zu 3, wobei 0 für Rohdaten und 3 für abgeleitete Informationen steht. Entlang dieser Skala befinden sich auch die verschiedenen Korrekturen und Prozessierungen, die die Ableitung der Information erst möglich machen und gleichzeitig ein Qualitätsmaß darstellen. Von rohen digitalen nummern (level 0) über radiometrisch und geometrisch korrigierte top-of-atmospheric (TOA) Produkte (Level 1B) zu zusätzlich ortho-referenzierten und räumlich geo-referenzierten Produkten (Level 1C) zu finalen Bottom-Of-Atmospheric (BOA) korrigierten Orthophotos, welche zusätzliche Bänder zur Aerosol Optical Thickness (AOT), Water Vapour (WV) und Quality Indicators (QI) besitzen. Bis jetzt werden Level 2 BOA Produkte nicht automatisch über die ESA produziert sondern müssen mittels spezieller Software wie der Sentinel Toolbox vom Nutzer manuell selbst produziert werden, was pro Satellitenbild etwa 10 Minuten benötigt und damit für die in der Arbeit verwendeten Satellitenbilder zeitliche unrealistisch ist. Obwohl die zusätzlichen Bänder des BOA Formats die Erstellung eines wolkenfreien Satellitenbild-Komposit vereinfachen würden, wird in der Arbeit mit den Level-1C Produkte gearbeitet, da sie das qualitativ hochwertigste Format darstellen, welches momentan in diesem Umfang zu Verfügung steht [8].

## 2.2. Percentil-Komposit

Die Perzentil-Methode ist im Gegensatz zur Greenest-Pixel-Methode sowohl einfacher als auch ressourcensparender. Sie zieht einfach aus einer Verteilung von Pixeln der selben Lage eine gewählte Stichprobenstatistik, hier das Perzentil  $P40$ . Da es innerhalb eines Jahres für ein Gebiet bis zu 60 sich überlagernde Satellitenbilder gibt, kann für jeden Pixel aus einer Verteilung gezogen werden.

Durch ziehen des  $P40$  werden besonders helle oder dunkle und damit seltene Werte, die z.B. durch Wolken oder Wolkenschatten entstehen automatisch aussortiert [1].

### *2.3. Greenest-Pixel-Komposit*

Die Erstellen eines Greenest-Pixel-Komposit folgt der Methodik von Stuhler et al. [9] mit dem Zusatz, dass als Vor-Prozessierung in den einzelnen Satellitenbildern die Wolken mit einem statistischen Klassifizierungs-Algorithmus erst detektieren und anschließend maskieren werden.

Bei der Erstellung eines Greenest-Pixel-Komposit wird zu Beginn, wenn noch nicht vorhanden, ein Qualitätsband zu jedem Satellitenbild hinzugefügt. Mithilfe dieses Qualitätsbandes wird Pixelweise entschieden welches Satellitenbild im finalen Mosaik verwendet wird. Das Satellitenbild, welches für einen gegebenen Pixel den höchsten Wert des Qualitätsbandes besitzt, steuert dabei die Werte aller Bänder für diesen Pixel bei.

#### *2.3.1. NDVI als Qualitätsband*

Für ein Greenest-Pixel-Komposit wird der Normalised-Difference-Vegetation-Index (NDVI) als Qualitätsband verwendet.

$$NDVI = \frac{NIR - Rot}{NIR + Rot}$$

Der NDVI reagiert sensibel auf Wolken und ist über wolkenfreier Vegetation deutlich höher als über Flächen welche von Wolken bedeckt sind. Bei vegetationsreichen Flächen ist der NDVI als Qualitätsband damit ein Index, der Verunreinigungen durch Wolken berücksichtigt und damit die Fläche automatisch in der vegetationsreichsten und wolkenlosesten Phase abbildet. Problematisch wird er für vegetationslose Oberflächen wie Fels, Sand und besonders Eis, Schnee oder Wasser [9].

#### *2.3.2. Wassermaske*

In Simbabwe dominiert die Trockensavanne, demnach können alle Vegetationslosen Oberflächen mit Ausnahme von Wasser vernachlässigt werden [10]. Um auch Wasser möglichst störungsfrei abzubilden, wurde für Wasserflächen das Perzentil  $P20$  als Qualitätsband verwendet. Zur spektralen Trennung von Wasser und Vegetation stellte sich der Kurzwellen-Infrarot Bereich von Band 12 der Sentinel 2 Satelliten als am geeignetsten heraus. Der NDWI (Normalized-Difference-Water-Index) erwies sich überraschenderweise als

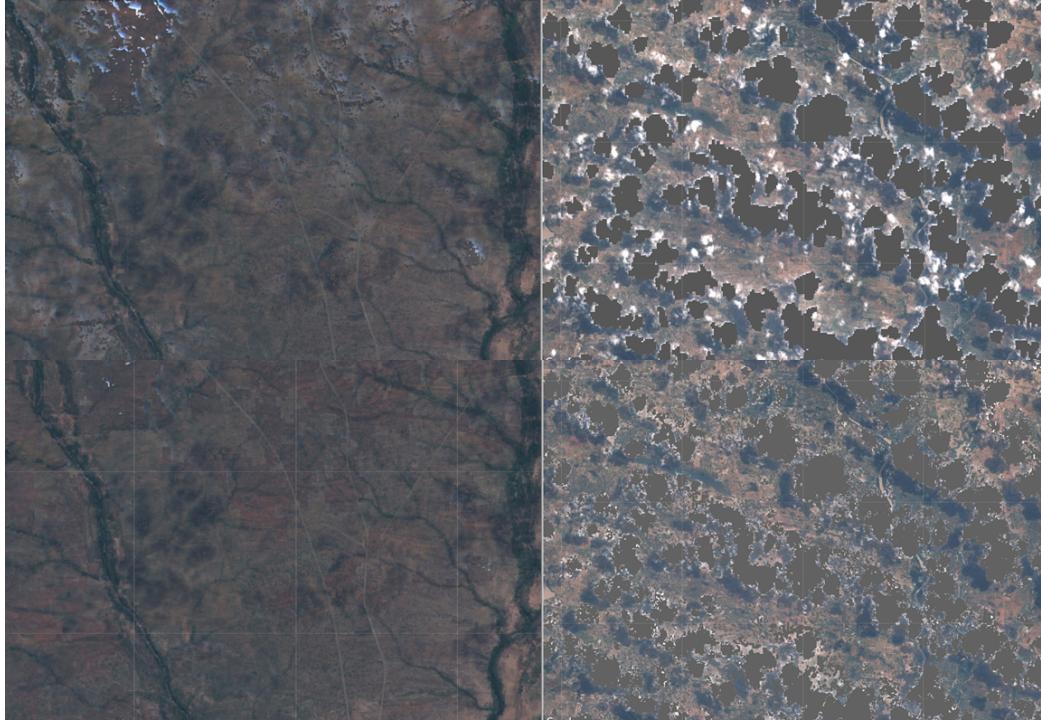


Abbildung 2: Greenest-Pixel-Komposit mit (unten links) und ohne (oben links) maskieren von Wolken, sowie Wolkenerkennung mit Band AQ60 (oben rechts) und Cart klassifizierer (unten rechts)

ungeeignet. Durch Band 12 (Kurzwellen-Spektrum des Infrarot) werden Wasserflächen von Vegetation unterschieden und ausgeschnitten (maskiert), im Anschluss wird die maskierte Fläche mit Pixeln eines Perzentil  $P20$  Komposit der selben Fläche ersetzt.

### *2.3.3. Detektieren von Wolken mit einem statistischen-Klassifizierung-Algorithmus*

Es hat sich gezeigt, dass sich die Störungen durch Wolken in einem Greenest-Pixel-Komposit deutlich verringert, wenn als Vor-Prozessierung die Wolken erst detektieren und dann in den Satellitenbildern maskiert werden (siehe Abbildung 2 links).

Für das detektieren von Wolken wurde ein Cart - und Random Forest Klassifizierer vergleichend genutzt. Die 12 Bänder der Sentinel Daten gingen dabei als Prädiktoren in das Model ein, als Antwortvariable wurde das zusätzliches Wolkenmaskenband (AQ60) genutzt. Beide Klassifizierer haben

eine fast identische Genauigkeit in der Vorhersage auf nicht gesehenen Daten (Accuracy berechnet aus der Fehlermatrix, RandomForest: 0.970, Cart: 0.978). Cart ist deutlich schneller und wird darum im weiteren verwendet.

Der AQ60 hat eine Auflösung von 60 m per Pixel und zeigt die Störung durch entweder Wolken, Wolkenschatten oder Cirrus Wolken an (siehe Abbildung 2 oben rechts). Der AQ60 ist, wie in Abbildung 2 rechts oben zu sehen, äußerst ungenau, kleinere Wolken und besonders Wolkenschatten sowie Cirrus-Wolken werden kaum eingefangen. Der Klassifizierer hingegen detektiert besonders Wolkenausläufer und Cirrus genauer (Abbildung 2, rechts unten) hat jedoch ebenfalls große Probleme mit Wolkenschatten, was durch die Trainingsdaten zu erklären ist (Garbage in = Garbage out). Auf diesen Punkt und mögliche Verbesserungen soll in der Diskussion eingegangen werden, wichtig ist hier, dass die Klassifikation eine bessere Detektierung der Wolken erlaubt als der AQ60. Der Trainingsdatensatz für den Klassifizierer wurde durch zufälliges ziehen von 30 000, Datenpunkten mit ausgeglichenen Klassen (Wolken, keine Wolken) aus den 1810 Satellitenbildern generiert (siehe Funktion 7.1.1). Mit diesen Trainingsdaten wurde der Cart Klassifizierer trainiert und auf jedes Satellitenbild angewendet. Bei jedem Bild wird ein zusätzlicher Wolken-Klassifikations-Band hinzugefügt, dieses Band wird anschließend verwendet um Flächen mit Wolken zu maskieren (siehe Skript 7.2).

### 3. Ergebnisse

Wie in Abbildung 3 zu sehen (oben Greenest-Pixel-Komposit, unten Perzentil-Komposit), ist es mit beiden Methoden möglich ein Wolkenfreies Satelliten-Komposit zu generieren. Abgesehen von dem leichten Rotstich des Perzentil-Komposit zeigen sich Unterschiede erst bei stärkerer Vergrößerung.

### 4. Diskussion

#### 4.1. Methoden Vergleich

Abbildung 4 zeigt ein vergrößerten Vergleich der Vegetationsabbildung von Greenest-Pixel-Komposit (links) und Perzentil-Komposit (rechts). Die Verwendung des NDVI als Qualitätsband im Greenest-Pixel-Komposit führt im Vergleich zum Perzentil-Komposit zu einer differenzierten Abbildung der Vegetation. Allerdings entstehen durch die nicht vollständige maskierung

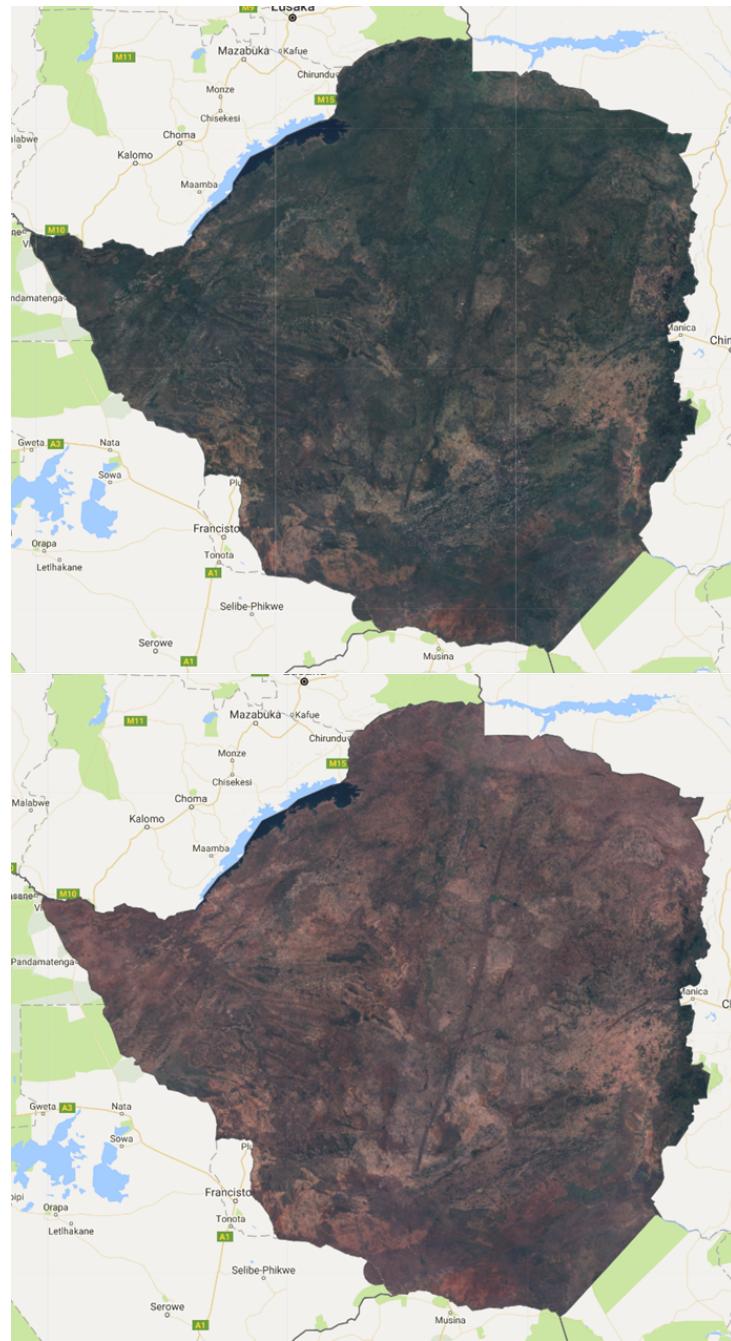


Abbildung 3: Wolkenfreies-Satelliten-Komposit mit einer Auflösung von 10 m pro Pixel.  
Vergleich von Greenest-Pixel-Komposit (oben) und des Perzentil-Komposit (unten)



Abbildung 4: Detaillierter Vergleich der Vegetationsabbildung von Greenest-Pixel-Kompositen (links) und dem Perzentil-Kompositen (rechts)



Abbildung 5: Cirrus-Artefakte und Wolkenschatten im Greenest-Pixel-Kompositen (links) und beinahe Störungsfreie Abbildung der selben Fläche im Perzentil-Kompositen (rechts)

von Wolken und besonders Wolkenschatten Artefakten welche durch Cirrus-Wolken hervorgerufen werden (siehe Abbildung 5).

Die Artefakte entstehen dadurch, dass der NDVI, welcher als Qualitätsband verwendet wird, zwar für dichtere Wolken sensibel ist, diese Sensibilität aber mit der Dichte der Wolken abnimmt und da nicht alle Wolken und besonders Cirrus von Klassifizierer detektiert und danach ausgeschnitten werden, gelangen diese teilweise in das finale Komposit. Das Perzentil-Komposit zeigt sich für alle Flächen als sehr robust im herausfiltern von sowohl Wolken als auch Wolkenschatten und besitzt damit kaum Artefakte. Jedoch ist die Perzentil-Methode nur sinnvoll für Gebiete in denen Wolken selten sind und dagegen äußerst problematisch für Gebieten mit sehr hoher Wolkendichte. Liegt die Häufigkeit der durch Wolken gestörten Pixeln über dem gewählten Perzentil werden Wolken im Perzentil-Komposit sichtbar. Obwohl also die Perzentil-Methode schneller und robuster ist, so kann sie das detektieren und ausschneiden von Wolken in gebieten mit hoher Wolkendichte nicht ersetzen.

#### *4.2. Verbesserung der Wolkendetektierung*

Der AQ60 als Antwortvariable für die Trainingsdaten ist unzureichend und wurde nur verwendet um das generieren von Trainingsdaten zu automatisieren. Kleinere Tests mit manuell generierten Trainingsdaten für einzelne Satellitenbildern haben gezeigt, dass sich die Detektierung von Wolken und Wolkenschatten durch einen Klassifizierer deutlich verbessern lässt. Ein Klassifizierer der nur mit wenigen, manuell generierte, Satellitenbilder trainiert wird, ist jedoch zu stark an diese Bilder angepasst ("overfitting"), es ist also aufgrund der Inhomogenität der Satellitenbilder notwendig das in den Trainingsdaten möglichst viele Bilder vertreten sind, was wiederum eine Automatisierung erfordert. Wichtig für eine Verbesserung der Genauigkeit des Klassifizierers ist dabei, dass Cirrus-Wolken und Wolkenschatten stärker in den Trainingsdaten vertreten sind. Um das zu erreichen könnte als neue Antwortvariable eine Kombination von AQ60 und Band 10, welches Cirrus abbildet, verwendet werden.

### **5. Schlussfolgerung**

Die Arbeit legt den Schluss nahe, dass beide Methode zur Erstellung eines wolkenfreien Satelliten-Komposit geeignet sind und dabei verschiedene Stärken und Schwächen besitzen. Die Perzentil-Methode hat sich als äußerst

robust für Gebiete mit schwacher bis Mittlerer Wolkendichte erwiesen und die Greenest-Pixel-Methode bietet eine detailliertere Abbildung der Vegetation und ist durch das Detektieren und maskieren von Wolken flexibler auch für Gebiete mit hoher Wolkendichte einzusetzen. Die Methode ist aber noch nicht ausgereift, ist noch Fehleranfällig und muss weiter verbessert werden.

Das Vorgehen, die Qualität eines wolkenfreien Satelliten-Komposit dadurch zu verbessern, dass als Vor-Prozessierung in den einzelnen Satellitenbildern die Wolken mit einem statistischen Klassifizierungs-Algorithmus erst detektieren und anschließend maskieren werden, hat sich bewehrt. Selbst mit unzureichenden Trainingsdaten war die Klassifizierung der Wolkenmaske in den Sentinel Daten überlegen. Dieses Vorgehen hat demnach Potential und sollte wie in der Diskussion vorgestellt weiter verbessert werden.

## 6. Bibliographie

- [1] C. Observer, Sentinel-2 cloudless mosaic the first global (almost) cloud-free view of the planet, 2017. <http://copernicus.eu/news/sentinel-2-cloudless-mosaic-first-global-cloud-free-view-planet>.
- [2] M. Li, S. C. Liew, L. K. Kwok, Automated production of cloud-free and cloud shadow-free image mosaics from cloudy satellite imagery, in: Proceedings of the XXth ISPRS Congress, pp. 12–13.
- [3] Google earth engine data catalog, 2017. <https://earthengine.google.com/datasets/>.
- [4] M. C. Hansen, P. V. Potapov, R. Moore, M. Hancher, S. Turubanova, A. Tyukavina, D. Thau, S. Stehman, S. Goetz, T. Loveland, et al., High-resolution global maps of 21st-century forest cover change, science 342 (2013) 850–853.
- [5] J.-F. Pekel, A. Cottam, N. Gorelick, A. S. Belward, et al., High-resolution mapping of global surface water and its long-term changes, Nature 540 (2016) 418–422.
- [6] Google earth engine case studies, 2017. [https://earthengine.google.com/case\\_studies/](https://earthengine.google.com/case_studies/).
- [7] S. Delwart, Sentinel-2 user handbook (2015).
- [8] Understanding sentinel-2 satellite data, 2014. <https://eox.at/2015/12/understanding-sentinel-2-satellite-data/>.
- [9] S. Stuhler, R. Leiterer, P. Joerg, H. Wulf, M. Schaepman, Generating a cloud-free, homogeneous landsat-8 mosaic of switzerland using google earth engine (2016).
- [10] Wikipedia, Simbabwe — wikipedia, die freie enzyklopädie, 2017. <https://de.wikipedia.org/w/index.php?title=Simbabwe&oldid=169236113>.

## 7. Anhang

### 7.1. Funktionen

#### 7.1.1. balancedSampling

```
exports.doc = 'sampleBalancedImage(image, geometry, pixel)' +  
  '\n Function to random sampling with balanced classes'  
  
exports.sampleBalancedImage = function(image1) {  
  // default bands  
  var bands = ['B2', 'B3', 'B4', 'B5', 'B6', 'B7', 'B8', 'B9',  
    'B10', 'B11', 'B12', 'class'];  
  // cloud band  
  var image = ee.Image(image1)  
  var cloud = image.select('QA60')  
  .unmask(0)  
  .remap([0, 1024, 2048], [0, 1, 1])  
  .rename('class')  
  // add cloud band  
  var imageClass = image.addBands(cloud)  
  // sample for noClouds  
  var imageSample0 = imageClass  
    .select(bands)  
    .updateMask(imageClass.select('class').eq(0))  
    .sample({numPixels: 1000})  
    .randomColumn('x')  
    .sort('x')  
    .limit(500)  
    .select(bands)  
  // sample for clouds  
  var imageSample1 = imageClass  
    .select(bands)  
    .updateMask(imageClass.select('class').eq(1))  
    .sample({numPixels: 1000})  
    .randomColumn('x')  
    .sort('x')  
    .limit(500)  
    .select(bands)  
  
  return imageSample0.merge(imageSample1)  
}
```

#### 7.2. Script to produce the cloud-free-composites

```
//////////
```

```

// define helper functions
///////////////////////////////
// function for classify all images in a collection
// adds classification band and smooth neighborhood

var classifyCloud = function(image) {
  var classification = ee.Image(image).classify(classifier)
//  var classificationSmooth = classification.
    reduceNeighborhood({
//    reducer: ee.Reducer.max(),
//    kernel: ee.Kernel.circle(2),
//  })
  return image.addBands(classification.rename('classification'))
}
;

// function to add NDVI
var addNdvi = function(image) {
  var ndvi = image.normalizedDifference(['B8', 'B4']).rename('ndvi')
  return image
    .addBands(ndvi)
};

// function to mask clouds with the classification band
var cloudMask = function(image) {
  return ee.Image(image).updateMask(image.select(['classification']).eq(0))
};

var waterMasc = function(mosaic, collection, B12Threshold) {
  var bands = ['B2', 'B3', 'B4', 'B5', 'B6', 'B7', 'B8', 'B9',
    'B10', 'B11', 'B12'];
  var mosaicSubBands = mosaic.select(bands)
  var collectionSubBands = collection.select(bands)
  var percentile = collectionSubBands.reduce(ee.Reducer.percentile([20]))
//  rename bands      .select(['B2_p20', 'B3_p20', 'B4_p20', 'B5_p20', 'B6_p20', 'B7_p20', 'B8_p20', 'B9_p20', 'B10_p20', 'B11_p20', 'B12_p20'],
  bands)
  return mosaicSubBands
    .where(percentile.select('B12').lt(ee.Number(B12Threshold)), percentile)
}
;
```

```

};

///////////////////////////////
//load data
///////////////////////////////

// boundary of zimbabwe
var zimbabwe = ee.FeatureCollection("USDOS/LSIB/2013")
  .filter(ee.Filter.stringContains('name', 'ZIMBABWE'))
var bands = ['B2', 'B3', 'B4', 'B5', 'B6', 'B7', 'B8', 'B9', ,
  B10', 'B11', 'B12'];

// load sentinel full data for zimbabwe
var trainingData = ee.ImageCollection('COPERNICUS/S2')
  .filterDate('2016-01-01', '2016-12-01')
  .filterBounds(zimbabwe)
// .filter(ee.Filter.gt('CLOUDY_PIXEL_PERCENTAGE', 5))
  .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 25))
  .limit(30)

// Band without class for classification
var bands = ['B2', 'B3', 'B4', 'B5', 'B6', 'B7', 'B8', 'B9', ,
  B10', 'B11', 'B12'];

var classificationData = ee.ImageCollection('COPERNICUS/S2')
  .filterDate('2016-01-01', '2016-12-30')
  .filterBounds(zimbabwe)
  .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 5))
//print(classificationData.size(), 'number of images')

///////////////////////////////
// balanced random sampling from image collection
///////////////////////////////

// load function to stratified random sampling
var ss = require('users/JesJehle/CloudFreeMosaic:Modules/
  sampleBalancedImage')
// sample from training data and creat sampled data
var training = trainingData
  .map(ss.sampleBalancedImage).flatten()

///////////////////////////////
// Train models
///////////////////////////////
//CART classifier with default parameters.

```

```

var classifier = ee.Classifier.cart().train(training, 'class'
, bands);

///////////////////////////////
// generate the final mosaic
///////////////////////////////

var cloudMaskedMosaic = classificationData
.map(addNdv)
.map(classifyCloud)
.map(cloudMask)
.qualityMosaic('ndvi')

///////////////////////////////
// generate percentile mosaic
///////////////////////////////

var percentileMosaic = classificationData.reduce(ee.Reducer.
percentile([40]))

// water mask
var cloudMaskedMosaicWater = waterMasc(cloudMaskedMosaic,
classificationData, 300).clip(zimbabwe)
var percentileMosaic = classificationData.reduce(ee.Reducer.
percentile([40])).clip(zimbabwe)

///////////////////////////////
// visualisations
///////////////////////////////

Map.addLayer(cloudMaskedMosaicWater, {bands: ['B4', 'B3', 'B2
'], min:0, max:3200}, 'mosaic with cloud masking');
Map.addLayer(percentileMosaic, {bands: ['B4_p40', 'B3_p40', ,
B2_p40], min:0, max:3200}, 'Percentile mosaic');

```