Analyse von thermalen Drohnenbildern

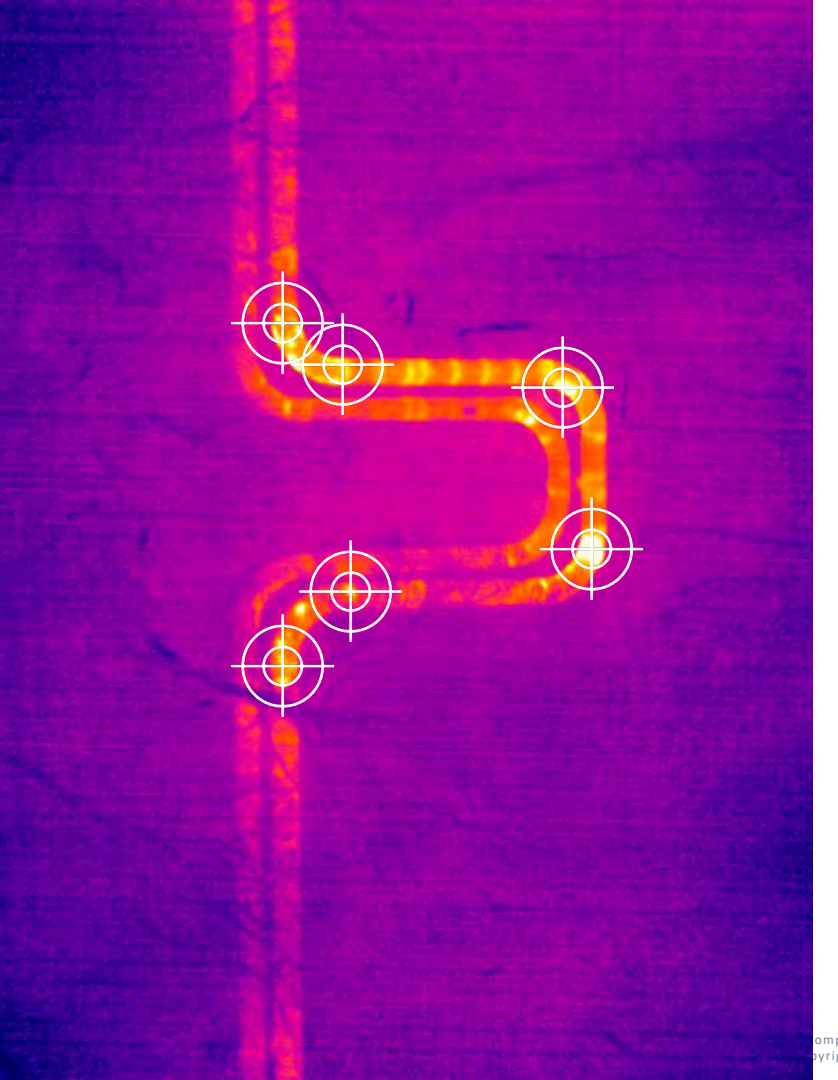
Von der Idee bis zur Umsetzung

Kurz zu mir

- Jannic Horst, 24 (Heidelberg, GER)
- Data Science Advisor
- Versicherungsteam
- Schwäche für Kameras

Das Projekt

Bildakquise



Bildakquise

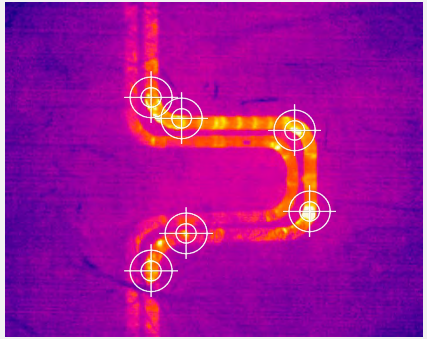
Händische Annotation

Thermaluntersuchung

Pipelinepark, 420 Rohrstr.

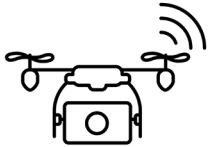
Messungen

1	35°C
2	44°C
3	23°C
4	32°C
5	37°C



Parameter

Emissionsgrad	1
Refl. Temp.	22°C
Abstand	20m
Temp. Extern	22°C



Bildakquise

Händische Annotation

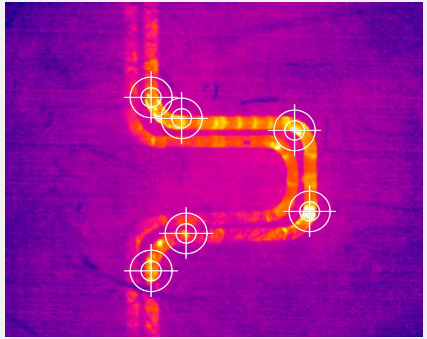
PDF-Report an Kunden

Thermaluntersuchung

Pipelinepark, 420 Rohrstr.

Messungen

1	35°C
2	44°C
3	23°C
4	32°C
5	37°C



Parameter

Emissionsgrad	1
Refl. Temp.	22°C
Abstand	20m
Temp. Extern	22°C

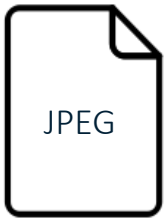
Bildakquise

Automatischen Analyse

PDF-Report an Kunden

Rahmenbedingungen

- Radiometrisches JPEG
- Proprietäres Format
- JPEG + Rohe Sensordaten
- Metadaten (GPS, Winkel, etc…)



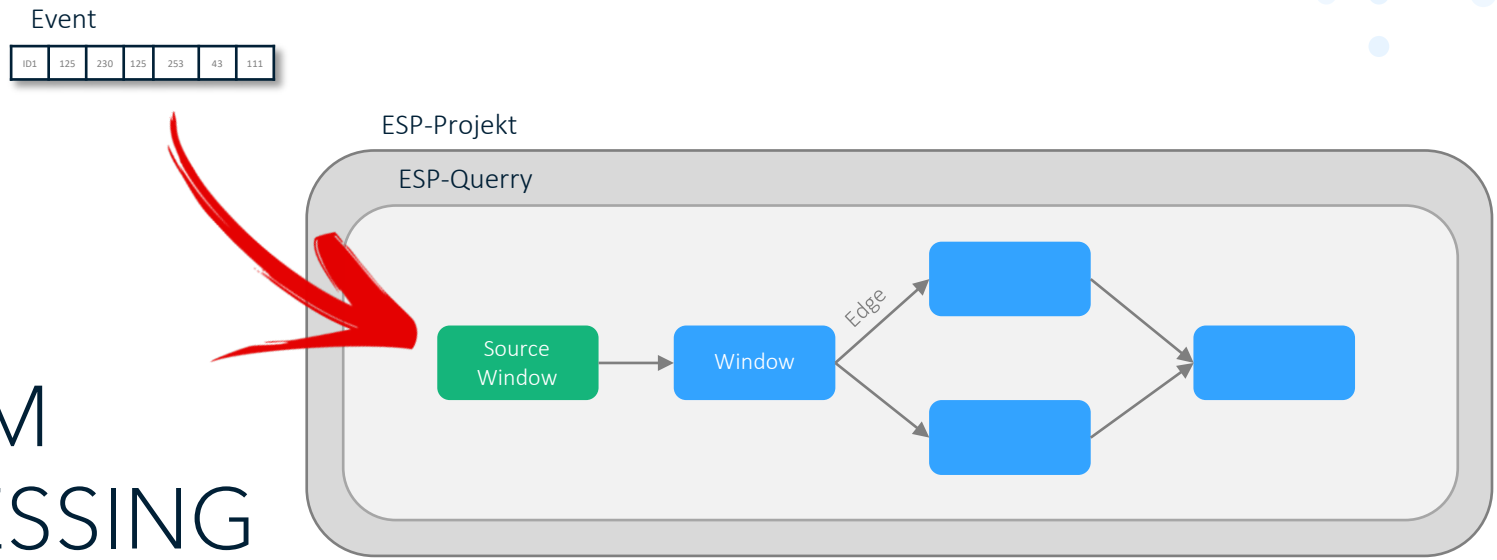
- 20MP Farbbild
- Metadaten (GPS, Winkel, etc…)

Jeweils ~200 Bilder

SAS
EVENT
STREAM
PROCESSING

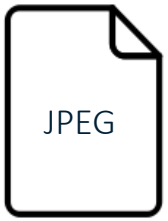
Als Plattform

SAS EVENT STREAM PROCESSING



Rahmenbedingungen

- Radiometrisches JPEG
- Proprietäres Format
- JPEG + Rohe Sensordaten
- Metadaten (GPS, Winkel, etc…)



- 20MP Farbbild
- Metadaten (GPS, Winkel, etc…)

Jeweils ~200 Bilder

SAS
EVENT
STREAM
PROCESSING

„dog“

BILD**KLASSIFIZIERUNG**

BILD**KLASSIFIZIERUNG**

OBJEKT ERKENNUNG

BILD**KLASSIFIZIERUNG**

OBJEKT ERKENNUNG

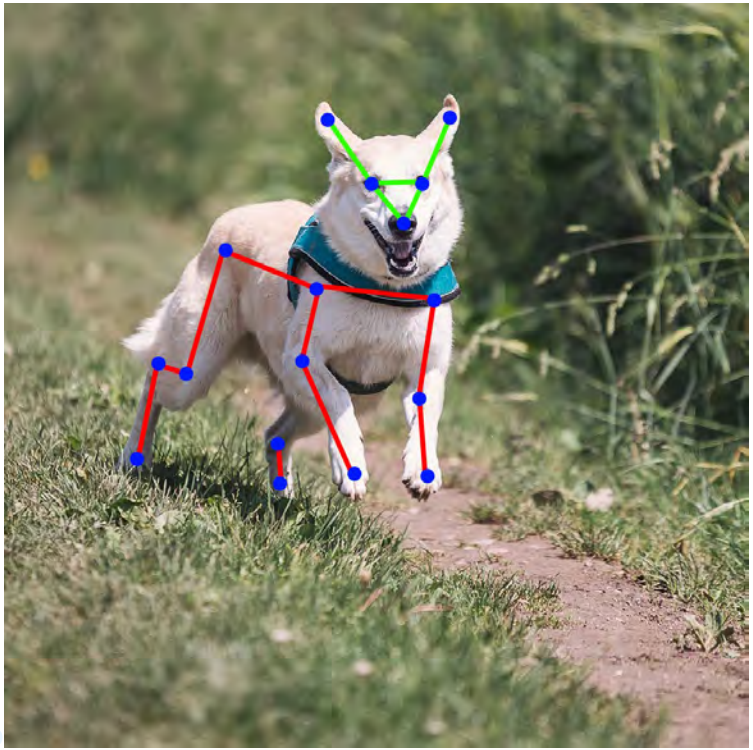
BILD**SEGMENTIERUNG**

BILD**KLASSIFIZIERUNG**

OBJEKT ERKENNUNG

BILD**SEGMENTIERUNG**

INSTANZSEGMENTIERUNG



BILD**KLASSIFIZIERUNG**

OBJEKT ERKENNUNG

BILD**SEGMENTIERUNG**

INSTANZSEGMENTIERUNG

KEYPOINT-ERKENNUNG

BILD**KLASSIFIZIERUNG**

OBJEKT ERKENNUNG

BILD**SEGMENTIERUNG**

INSTANZSEGMENTIERUNG

KEYPOINT-ERKENNUNG

BILD-**MATCHING**

White dog with a blue collar runs on a field trail

BILD**KLASSIFIZIERUNG**

OBJEKT ERKENNUNG

BILD**SEGMENTIERUNG**

INSTANZSEGMENTIERUNG

KEYPOINT-ERKENNUNG

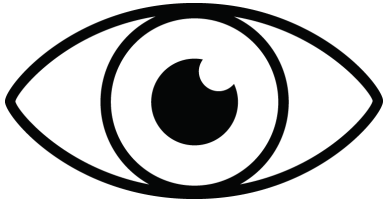
BILD-**MATCHING**

BILD**BESCHRIFTUNG**

BILD**SEGMENTIERUNG**

INSTANZSEGMENTIERUNG

OBJEKT ERKENNUNG



KEYPOINT ERKENNUNG

BILD**KLASSIFIZIERUNG**

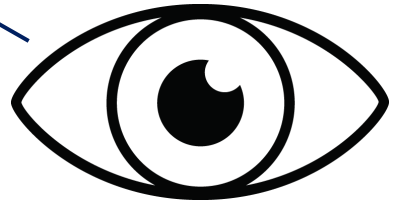
BILD-**MATCHING**

BILD**BESCHRIFTUNG**

BILD**SEGMENTIERUNG**

INSTANZSEGMENTIERUNG

OBJEKT ERKENNUNG

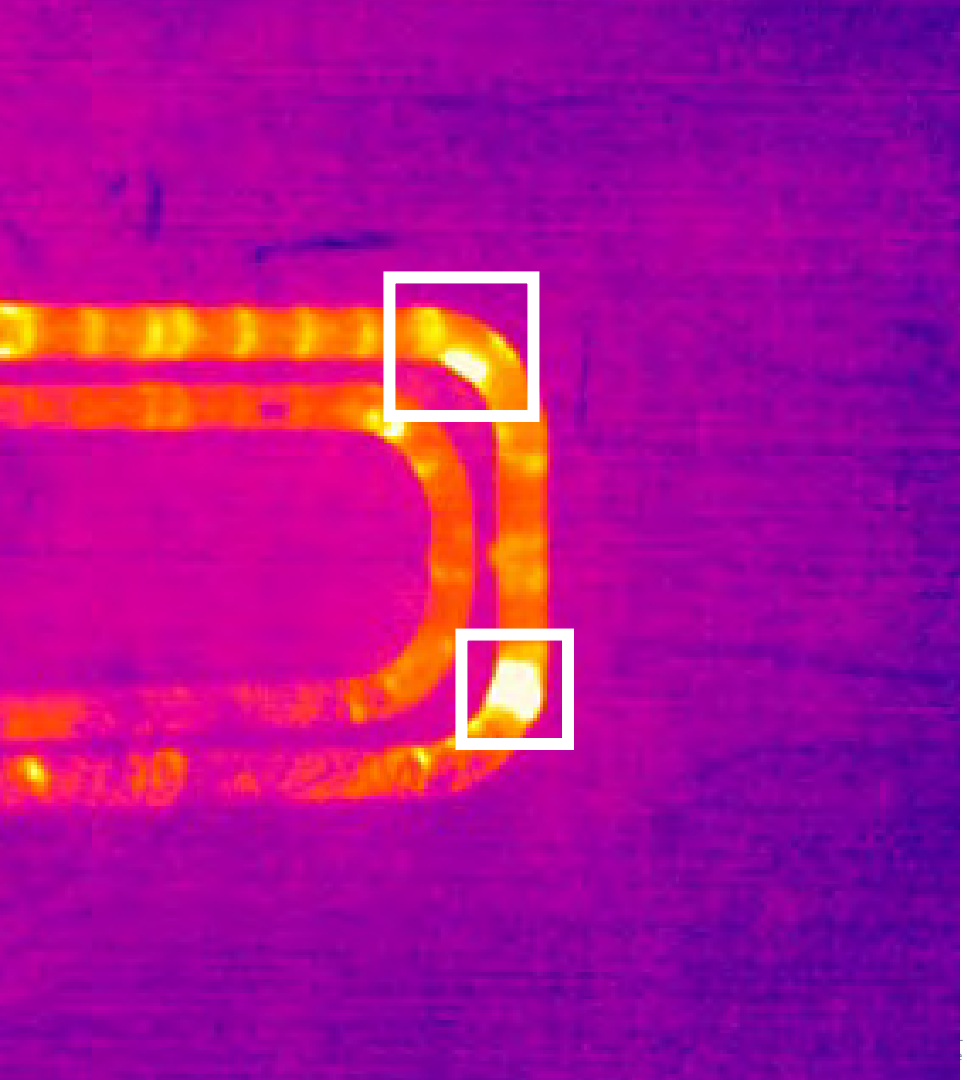


KEYPOINT ERKENNUNG

BILD**KLASSIFIZIERUNG**

BILD-**MATCHING**

BILD**BESCHRIFTUNG**

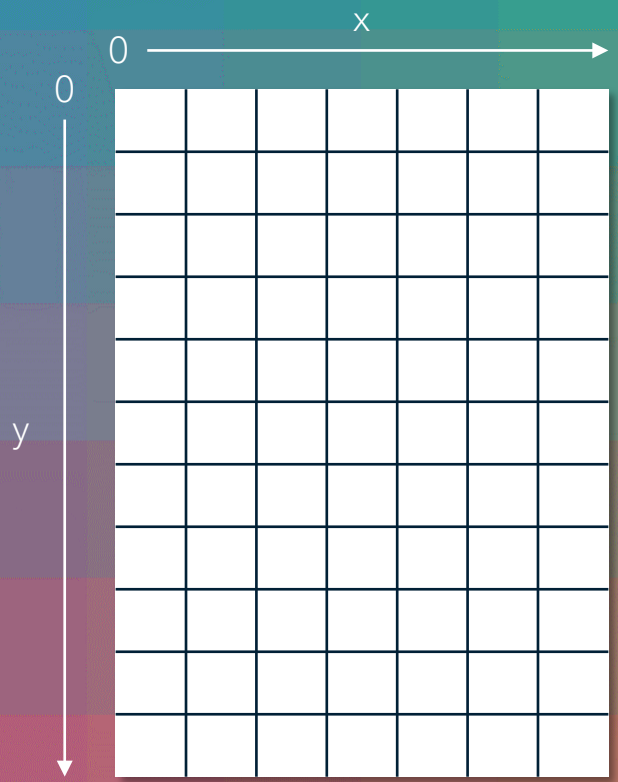


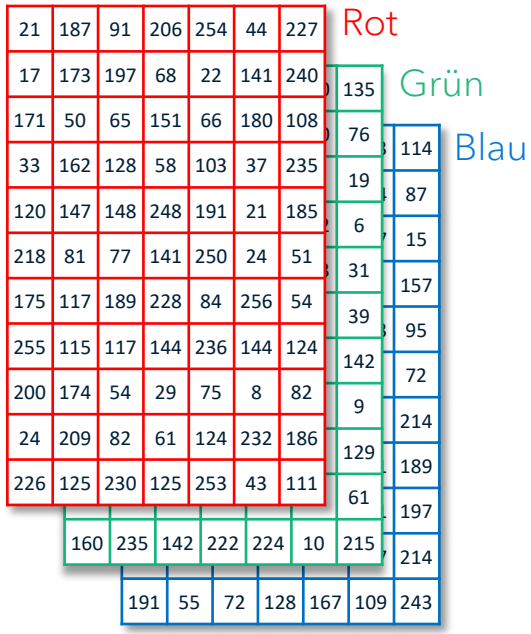
WARUM KEINE **NEURONALEN NETZE?**

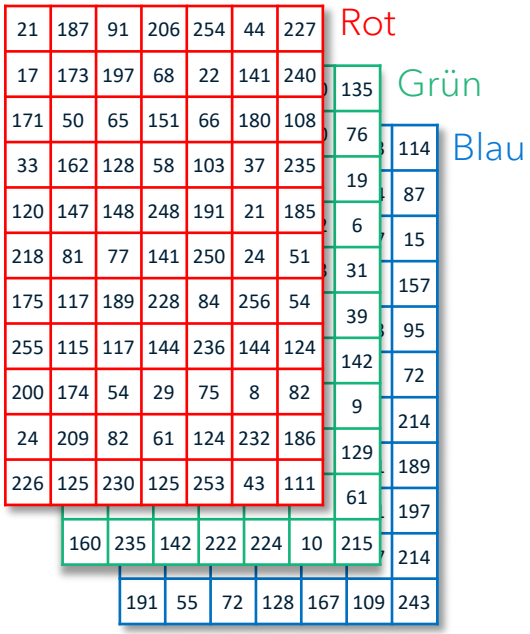
1. Variabilität der Hotspotgröße
2. Inkonsistenz der Form
3. **Zu wenig Trainingsdaten**

Ein neuer Ansatz

1. Behandle Pixel als einzelne Datenpunkte auf einer Ebene


x
0
0
y





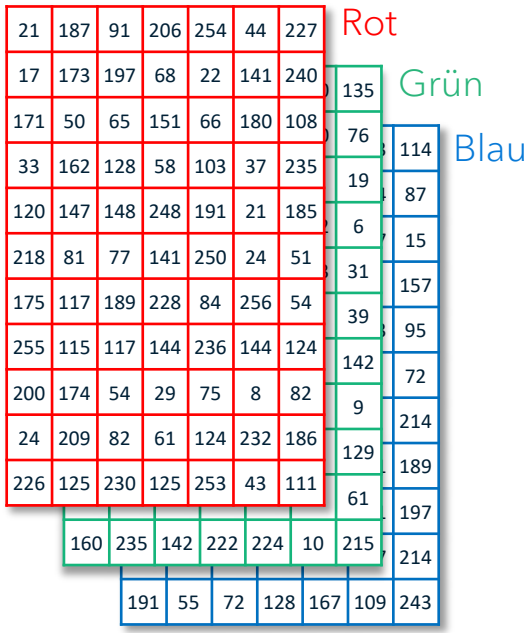
R:**255** G:**0** B:**0**

R:**1** G:**32** B:**54**

R:**21** G:**181** B:**123**

R:**255** G:**204** B:**50**

$2^8 \times 2^8 \times 2^8 \approx 16.7M$ Farben



21	187	91	206	254	44	227
17	173	197	68	22	141	240
171	50	65	151	66	180	108
33	162	128	58	103	37	235
120	147	148	248	191	21	185
218	81	77	141	250	24	51
175	117	189	228	84	256	54
255	115	117	144	236	144	124
200	174	54	29	75	8	82
24	209	82	61	124	232	186
226	125	230	125	253	43	111

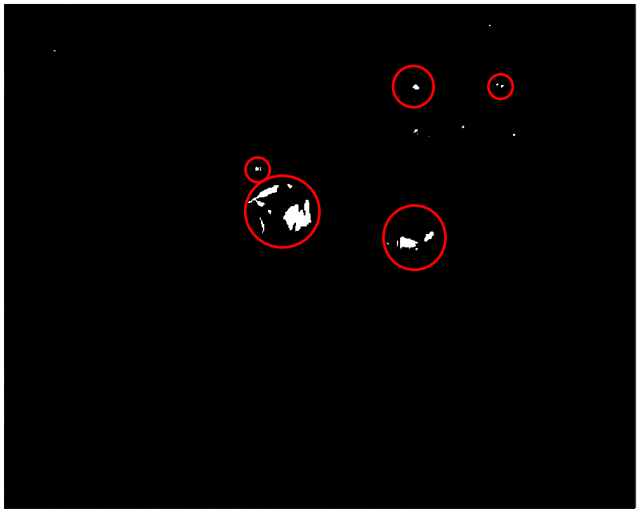
10	35	23	25	7	5	24
22	22	17	23	9	10	13
31	5	11	27	24	18	15
28	24	31	7	6	13	21
15	13	11	13	29	12	13
21	15	25	11	11	14	31
33	13	17	26	8	17	22
34	26	15	34	28	7	5
13	18	9	27	33	20	24
33	22	13	23	12	9	15
19	27	15	33	27	28	18

„Temperatur“

Ein neuer Ansatz

1. Behandle Pixel als einzelne Datenpunkte auf einer Ebene
2. Versuche nach Pixeln zu filtern die (vermutlich) zu Hotspots gehören
3. Gruppiere diese Pixel zu Hotspots

Ein neuer Ansatz



1. Filterung Binär-Schwellenwert
2. Clustering der Pixel ≠ 0
3. Regelbasierte Filterung der identifzierten Cluster

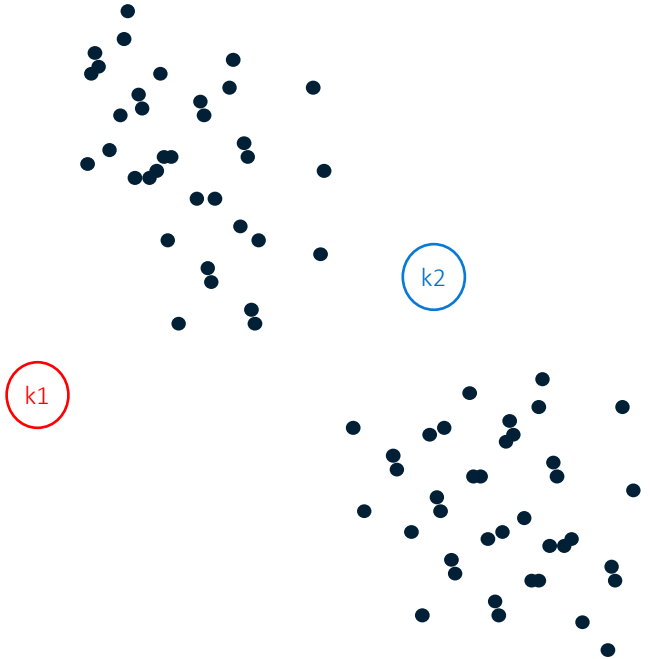

THRESHOLDING
+CLUSTERING

k = 2

DER KLASSIKER: **K-MEANS CLUSTERING**

1. Wähle eine Anzahl von Clustern (k)

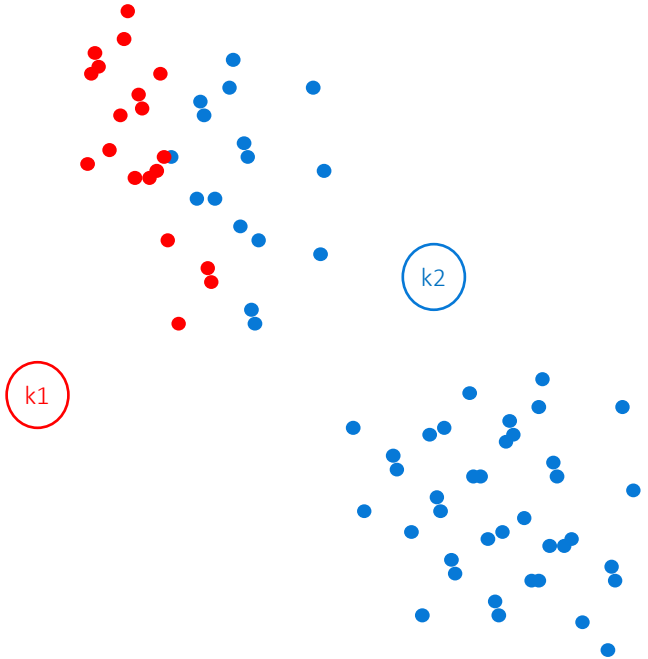
K = 2



DER KLASSIKER:
K-MEANS CLUSTERING

1. Wähle eine Anzahl von Clustern (k)
2. Platziere die Zentren aller Cluster zufällig
3. Wiederhole Schritt 4 & 5 bis Cluster optimiert sind

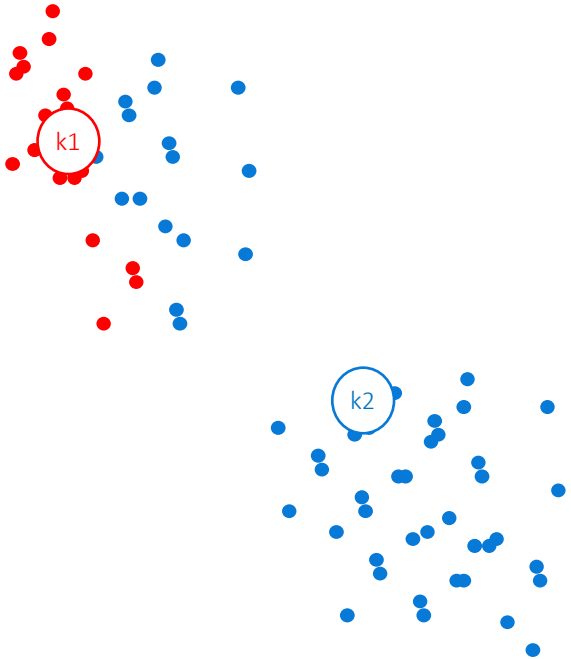
K = 2



DER KLASSIKER:
K-MEANS CLUSTERING

1. Wähle eine Anzahl von Clustern (k)
2. Platziere die Zentren aller Cluster zufällig
3. Wiederhole Schritt 4 & 5 bis Cluster optimiert sind
4. Finde zu jedem Datenpunkt das nächste Zentrum und weise ihm diesem Cluster zu

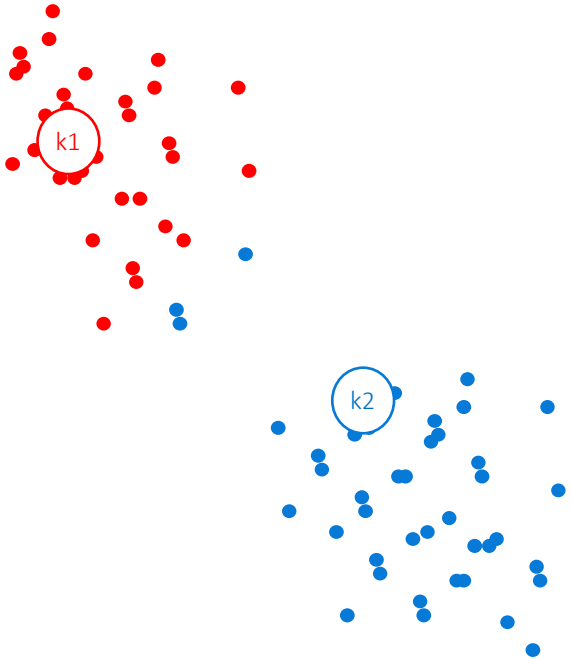
K = 2



DER KLASSIKER: **K-MEANS CLUSTERING**

1. Wähle eine Anzahl von Clustern (k)
2. Platziere die Zentren aller Cluster zufällig
3. Wiederhole Schritt 4 & 5 bis Cluster optimiert sind
4. Finde zu jedem Datenpunkt das nächste Zentrum und weise ihm diesem Cluster zu
5. Platziere die Zentren in die geometrische Mitte aller ihm zugewiesener Datenpunkte

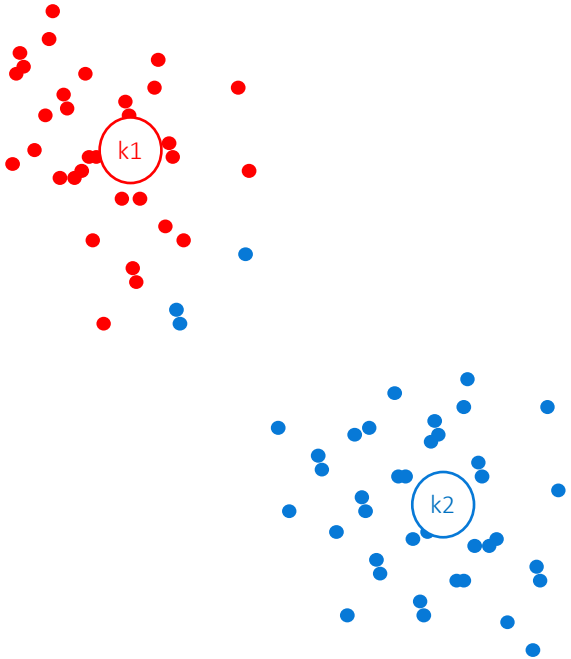
K = 2



DER KLASSIKER: **K-MEANS CLUSTERING**

1. Wähle eine Anzahl von Clustern (k)
2. Platziere die Zentren aller Cluster zufällig
3. Wiederhole Schritt 4 & 5 bis Cluster optimiert sind
4. **Finde zu jedem Datenpunkt das nächste Zentrum und weise ihm diesem Cluster zu**
5. Platziere die Zentren in die geometrische Mitte aller ihm zugewiesener Datenpunkte

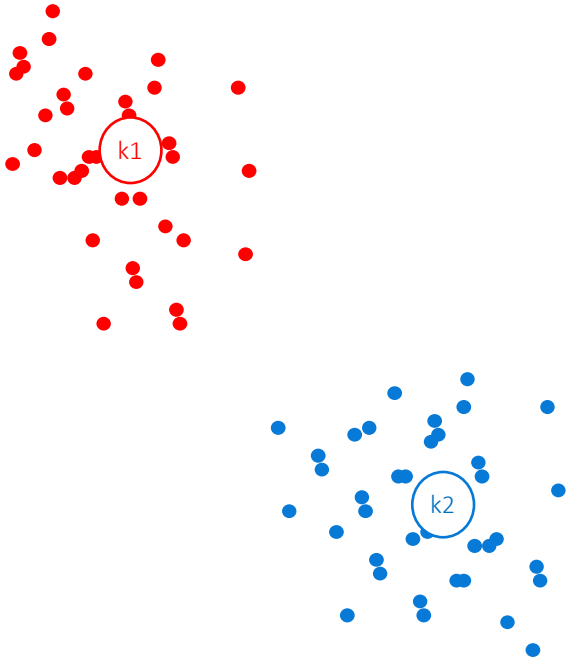
K = 2



DER KLASSIKER: **K-MEANS CLUSTERING**

1. Wähle eine Anzahl von Clustern (k)
2. Platziere die Zentren aller Cluster zufällig
3. Wiederhole Schritt 4 & 5 bis Cluster optimiert sind
4. Finde zu jedem Datenpunkt das nächste Zentrum und weise ihm diesem Cluster zu
5. Platziere die Zentren in die geometrische Mitte aller ihm zugewiesener Datenpunkte

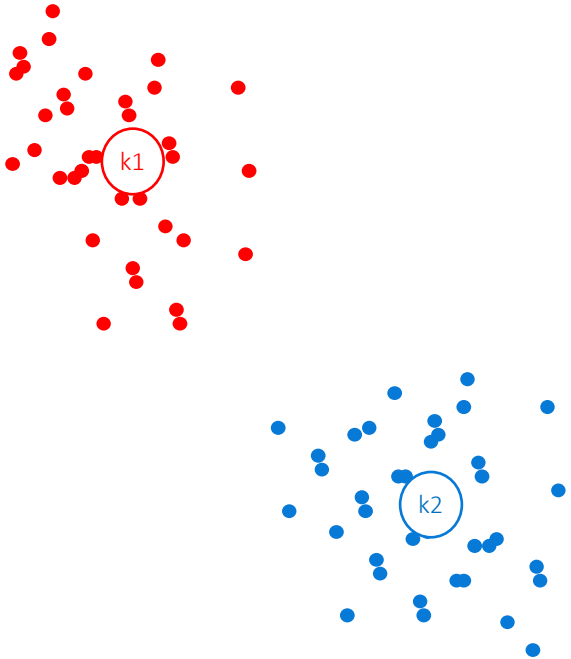
K = 2



DER KLASSIKER: **K-MEANS CLUSTERING**

1. Wähle eine Anzahl von Clustern (k)
2. Platziere die Zentren aller Cluster zufällig
3. Wiederhole Schritt 4 & 5 bis Cluster optimiert sind
4. **Finde zu jedem Datenpunkt das nächste Zentrum und weise ihm diesem Cluster zu**
5. Platziere die Zentren in die geometrische Mitte aller ihm zugewiesener Datenpunkte

K = 2



DER KLASSIKER: **K-MEANS CLUSTERING**

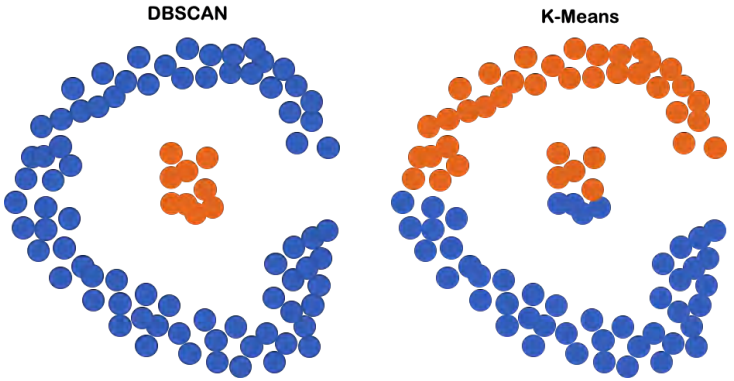
1. Wähle eine Anzahl von Clustern (k)
2. Platziere die Zentren aller Cluster zufällig
3. Wiederhole Schritt 4 & 5 bis Cluster optimiert sind
4. Finde zu jedem Datenpunkt das nächste Zentrum und weise ihm diesem Cluster zu
5. Platziere die Zentren in die geometrische Mitte aller ihm zugewiesener Datenpunkte

DAS PROBLEME MIT:
K-MEANS CLUSTERING

1. Anzahl der Cluster muss vor beginn festgelegt werden

2. Initiale Platzierung der Zentren beeinflusst das Ergebnis

3. Tendiert zu kreisförmigen Clustern



Was sind Alternativen?

DENSITIY **B**ASED **SCAN**NING

CONNECTED COMPONENT

\+ Besser darin **nicht-kreisförmige Cluster** zu erkennen

\+++ **Anzahl** der Cluster muss **nicht festgelegt** werden

Nochmal als Recap

1. **Filterung der Pixel auf Basis der Temperatur**

2. **Clustering aller Pixel ≠ 0 über die Koordinaten**
   - K-Means
   - DBScan
   - Connected Component

3. **Regelbasierter Ausschluss von Cluster zur Bereinigung**
   - Größe
   - Max. Temperatur

4. **Visualisierung**

Was kam dabei raus…

RJPEG

- Algorithmus der Wahl
- Schwellenwert-Methode
- Filter-Parameter

TXT

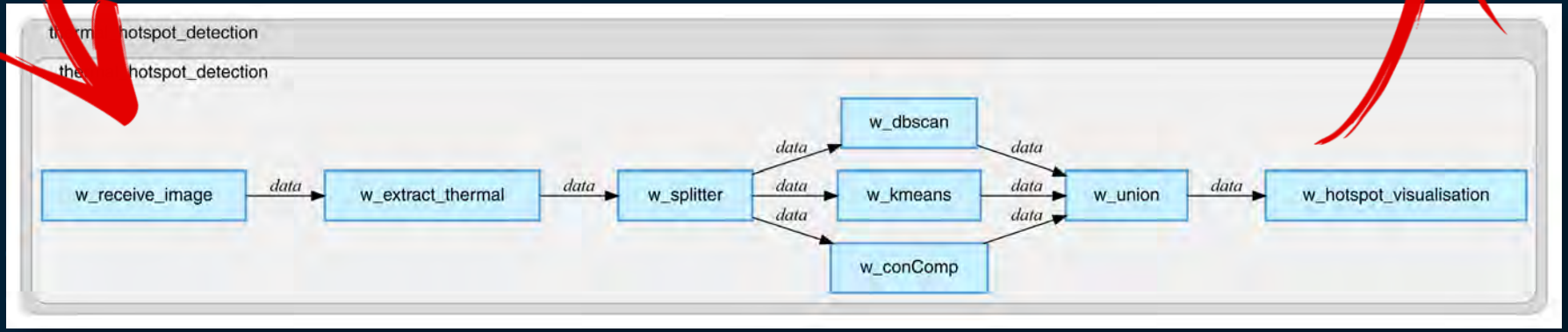
Ergebnisse

Jupyter Notebook

Publisher

ESP Server

Subscriber



SAS ESP ESPPy

WIE GUT HAT DAS AM ENDE **FUNKTIONIERT?**

93%

Der Annotationen wurden mit dem DBSCAN-Algorithmus erkannt

Ergebnisse in Zahlen

Algorithmus	Ø IoU (%) ↑	Überlappung* (%) ↑	Anzahl Hotspots** (%) ↑	Ø Höchstemperatur** (%) ↑
K-Means	0.02	1.00	1.88	1.29
DB-Scan	0.17	0.93	2.74	1.33
Connected Component	0.29	0.84	1.81	1.15

best 2nd best 3rd

* mit der Ground Truth

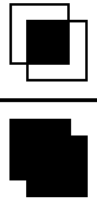
** im Verhältnis mit der Ground Truth

Ergebnisse in Zahlen

Algorithmus	Ø IoU (%) ↑	Überlappung* (%) ↑	Anzahl Hotspots** (%) ↑	Ø Höchstemperatur** (%) ↑
K-Means	0.02	1.00	1.88	1.29
DB-Scan	0.17	0.93	2.74	1.33
Connected Component	0.29	0.84	1.81	1.15

best 2nd best 3rd

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$



* mit der Ground Truth

** im Verhältnis mit der Ground Truth

Ergebnisse in Zahlen

Algorithmus	Ø IoU (%) ↑	Überlappung* (%) ↑	Anzahl Hotspots** (%) ↑	Ø Höchstemperatur** (%) ↑
K-Means	0.02	1.00	1.88	1.29
DB-Scan	0.17	0.93	2.74	1.33
Connected Component	0.29	0.84	1.81	1.15

best 2nd best 3rd

* mit der Ground Truth

** im Verhältnis mit der Ground Truth

VIELEN DANK!

jannic.horst@sas.com

@JannicHorst

All Things Data

linktr.ee/mltips

sas

Erstelle ESP-Projekt

```
esp = esppy.ESP(hostname='http://localhost:9900')
esp_project = esp.create_project('thermal_hotspot_detection',
                                 n_threads=1,
                                 pubsub='manual')
esp_project.add_continuous_query('thermal_hotspot_detection')
```

Erstelle Source-Window

```
w_receive_image = esp.SourceWindow(name='w_receive_image',
                                   schema=('id*:int64', 'filename:string',
'save_path:string','image:blob','threshold:int32','algorithm:string', 'min_size:int32',
'detection_thr:int32', 'lower_limit:int32'),
                                   index_type='empty',
                                   insert_only=True,
                                   autogen_key=True,
                                   pubsub=True)
esp_project.add_window(w_receive_image, contquery='thermal_hotspot_detection')
```

Verknüpfung von Windows

```
w_receive_image.add_target(w_extract_thermal, role='data')
w_extract_thermal.add_target(w_splitter, role= 'data')
```

Publisher

```
# create a list of images in a specific location
imgPath = glob.glob("/data/notebooks/testdata/v2/images/*")

#create the publisher
w_image_path_pub = w_receive_image.create_publisher(blocksize=1, rate=1, pause=0, opcode='insert',
                                                    format='csv')
#go through all images in the list
for img in imgPath:
    #set the input parameters
    _,filename = ntpath.split(img)
    save_path = '/data/notebooks/save_path/'
    algorithm = 'kmeans'
    threshold = -2
    min_size = 20
    detection_thr = 75
    lower_limit = 35

    #open image and encode it to Base64
    with open(img, 'rb') as binary_file:
        binary_image = binary_file.read()
        encoded_image = base64.b64encode(binary_image)

    #publish the message as string
    w_image_path_pub.send('i,n,1,"{}","{}","{}","{}","{}","{}","{}","{}",\n'.format(filename, save_path,
         encoded_image.decode(), threshold, algorithm, min_size, detection_thr,
         lower_limit))
```

Erstelle Subscriber

```
#create the schema for the data export
expo_schema_results = expoSchema(w_hotspot_visualisation, ('image', 'thermal_image',
                                                           'thermal_image_annotated'))

#get messages from visualisation window
def on_event(_, event):
    global event_received
    event_received = event

    #write event to csv file
    write_results(event_received, expo_schema_results)

#create the subscriber and start it
w_hotspot_visualisation_s = w_hotspot_visualisation.create_subscriber(on_event=on_event, pagesize=1000)
w_hotspot_visualisation_s.start()
```