Gmail **Kiyarash Farivar <kiyarashfarivar@gmail.com>**

Pipes dataset

21 messages

Kiarash Farivar <kiyarashfarivar@gmail.com> 18 March 2020 at 20:00
To: adrien.gressin@heig-vd.ch
Cc: Salzmann Mathieu <mathieu.salzmann@epfl.ch>, Remelli Edoardo <edoardo.remelli@epfl.ch>

Hello

I am the master student who's working on the pipe pointcloud segmentation project at epfl, and I have some questions regarding the data:

1. As mentioned earlier by Mathieu we need bounding boxes for the algorithm to work. You mentioned using euclidean clustering but that is not applicable since such an algorithm wouldn't know the difference between different instances of the same type of pipe and since pipes can be close to each other they are not distinguishable for such an algorithm. The bounding boxes we discuss are usually defined for each instance by their centre (x,y,z) the dimensions(height, width, length) and the angel of the box in the bird's eye view (x-z plane) and if we want something more accurate the angle with the other axis y. Is there any chance you could provide such data ?
(As a side note: If it is possible to put each pipe in it's own bounding box then there is also no need to classify the pointclouds by hand (which I guess was the case for you) you just provide a label for the bounding box (e.g. in a text file) and every point in the box is assumed to belong to that class of pipe, which could make the labelling job easier)

2. Color information is useful to us and looking at the original point clouds (not the subsampled version) the colors looked accurate but in the subsampled version they looked modified and everything is green. Could you correct the colors in the subsampeled ? (are other parameters like the reflection of each point accurate in the subsampled version ?).

3. You have this calibration information and for each image a location and angle information. Is it possible for you to provide us with a formula to map the image pixels into the corresponding points in the lidar coordinates using this data ?

4. Some of the folders you provided contained multiple pointclouds (more than two; for instance the folder 02_athenaz has 4) and multiple image folders. And the extra files have iphone in their names. What is their difference ? I think your final point cloud labelling results are always in the file that contains the word "subsampled" right ? and the other set of photos are the same scene but the photo is from an iphone but then why two folders "photo_iphone_img" and "photo_iphone_video" ?

Thank you for your time,
Kiarash Farivar

Adrien Gressin <adrien.gressin@heig-vd.ch> 19 March 2020 at 13:10
To: Kiarash Farivar <kiyarashfarivar@gmail.com>
Cc: Salzmann Mathieu <mathieu.salzmann@epfl.ch>, Remelli Edoardo <edoardo.remelli@epfl.ch>

Hi Kiarash,

my answers in the mail bellow :

On 18.03.20 20:00, Kiarash Farivar wrote:

> Hello
>
> I am the master student who's working on the pipe pointcloud segmentation project at epfl, and I have some questions regarding the data:
>
> 1. As mentioned earlier by Mathieu we need bounding boxes for the algorithm to work. You mentioned using euclidean clustering but that is not applicable since such an algorithm wouldn't know the difference between different instances of the same type of pipe and since pipes can be close to each other they are not distinguishable for such an algorithm. The bounding boxes we discuss are usually defined for each instance by their centre (x,y,z) the dimensions(height, width, length) and the angel of the box in the bird's eye view (x-z plane) and if we want something more accurate the angle with the other axis y. Is there any chance you could provide such data ?
> (As a side note: If it is possible to put each pipe in it's own bounding box then there is also no need to classify the pointclouds by hand (which I guess was the case for you) you just provide a label for the bounding box (e.g. in a text file) and every point in the box is assumed to belong to that class of pipe, which could make the labelling job easier)

In my opinion, most of instances are well separated, and it would be easier to proceed like this, and to manually correct the few remaing errors.

Unfortunately, since we have a lot of work for creating online lectures for ours students, I will not have time to spend on this. I think it would be a good programming exercise for you (It doesn't seem like a very complicated thing to do for an EPLF master student ;))

> 2. Color information is useful to us and looking at the original point clouds (not the subsampled version) the colors looked accurate but in the subsampled version they looked modified and everything is green. Could you correct the colors in the subsampeled ? (are other parameters like the reflection of each point accurate in the subsampled version ?).

I don't know which software you are using to read las file, but every las file I provide to you contain RGB (color) data,

Some pointclouds, the "classified" ones contain a label for each point. Maybe your las viewer color such data using classification data and not color data, you may modified it on your viewer setting.

If you use a python code to read such data (for exemple using liblas), you may have access for each point to the classification value (p.classification) et the color value (p.color)

--> https://liblas.org/tutorial/python.html

> 3. You have this calibration information and for each image a location and angle information. Is it possible for you to provide us with a formula to map the image pixels into the corresponding points in the lidar coordinates using this data ?

You will find a pdf in attachment that show how to compute image position (x,y) from world position (X,Y,Z), for that you will need :

- cameras distortion parameters f, cx, cy, ki, pi, ... (for example in file 01_jussy2/photo_sony/Jussy2_calib_cam.xml)

- cameras position and orientation X,Y,Z, O,P,K (for example in file 01_jussy2/photo_sony/Jussy2_opk.txt)

then you will be able to apply the formula given in the attached pdf.

> 4. Some of the folders you provided contained multiple pointclouds (more than two; for instance the folder 02_athenaz has 4) and multiple image folders. And the extra files have iphone in their names. What is their difference ? I think your final point cloud labelling results are always in the file that contains the word "subsampled" right ? and the other set of photos are the same scene but the photo is from an iphone but then why two folders "photo_iphone_img" and "photo_iphone_video" ?

Since one aspect of this project is to compare the ability of various sensors to map underground objects, we provide you, for each areas several point cloud :

- the reference one, acquired using a Leicia RTC360 lidar, **which is the only labelled point cloud** (for example 01_jussy2/clouds/Jussy2_rtc360_subsampled_classified.las)

- one (or more) point cloud computed from images, ie by photogrammetry process (name XXX_photo_XXX.las), which are not labelled (you must use labelled from the reference point cloud)

Such photogrammetric point clouds, have been computing using various cameras (not every cameras have been use on every areas.) :

- Sony (full frame / high value camera) --> name XXX_photo_sony.las

- Iphone in picture mode --> name XXX_photo_iphone_ing.las

- Iphone in video mode --> name XXX_photo_iphone_video.las

- Iphone using the application "kickthemap" --> name XXX_photo_ktm.las

Each photogrammetric point cloud is coming with a folder containing raw image, camera calibration file, and cameras position and orientation files,

For example, in Jussy1 area, you will found :

- the reference point cloud : 01_jussy1/clouds/Jussy1_rtc360_subsampled_classified.las

- the photogrammetric point cloud from Sony camera : 01_jussy1/clouds/Jussy1_photogra_sony.las

- the corresponding photogrammetric data in : 01_jussy1/photo_sony, ie :

    - cameras position and orientation file : 01_jussy1/photo_sony/Jussy1_opk.txt

    - camera calibration file : 01_jussy1/photo_sony/Jussy1_calib_cam.xml

    - raw images in the folder JPG

> Thank you for your time,
> Kiarash Farivar

Hope such information will be helpfull for you, please tell me if you need more details or if some part are not clear enought.

Best regards,

Adrien Gressin

Agisoft_rotation_distortion.pdf
28K

Kiarash Farivar <kiyarashfarivar@gmail.com> 25 March 2020 at 11:54
To: Adrien Gressin <adrien.gressin@heig-vd.ch>

Thank you for the answers it cleared up some of my confusions. I have some more questions:

1. what application do you use for visualizing the .las files ?

2. Where does the color information in the Leicia .las file come from? (is it from the images already provided ? or ...)

about the transformations:

3. So the two .las files we have one is from the Leicia and the other is constructed from the images (using photogrammetry).
How can I transform a location in the photogrammetry point cloud coordinates into a location in the Leicia point cloud coordinates?

4. we need to know how each **pixel** of an image maps to a location in the **Leicia lidar** coordinates. How can I do this ?
what you provided seems to be transformation from the **photogrammetry** coordinates (X,Y,Z) to the **image position** (x,y).
also what does image position (x,y) mean? (what coordinates do we use ? where is the origin (0,0) ?)

Thank you for your help,
Kiarash
[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 25 March 2020 at 12:25
To: Kiarash Farivar <kiyarashfarivar@gmail.com>

Hi Kiarash,

You will find my answers below in your mail

Regards,

Adrien

On 25.03.20 11:54, Kiarash Farivar wrote:

> Thank you for the answers it cleared up some of my confusions. I have some more questions:
>
> 1. what application do you use for visualizing the .las files ?

You may use CloudCompare, which is free and opensource.

> 2. Where does the color information in the Leicia .las file come from? (is it from the images already provided ? or ...)
>
> about the transformations:

Leica color information come from internal sensor cameras, but we don't have access to such images (we may have such data using the C# API, but we need some dev to do it)

> 3. So the two .las files we have one is from the Leicia and the other is constructed from the images (using photogrammetry).
> How can I transform a location in the photogrammetry point cloud coordinates into a location in the Leicia point cloud coordinates?

All data are in the same coordinate system, so XYZ_photogrammetric = XYZ_leica

> 4. we need to know how each **pixel** of an image maps to a location in the **Leicia lidar** coordinates.
> How can I do this ?
> what you provided seems to be transformation from the **photogrammetry** coordinates (X,Y,Z) to the **image position** (x,y).
> also what does image position (x,y) mean? (what coordinates do we use ? where is the origin (0,0) ?)

For what I understand from the previous work (on CFF data), you may compute for each leica lidar point, the NN feature centered on the the corresponding pixel on the image, so for me you need the (X,Y,Z) to image positon (x, y), isn't it ?

If not, the other transformation is not possible for two reason :

- there is not a lidar point for each image pixel (you may have so hidden part)

- even if we suppose the latter, you may need to know the depth (distance from the camera to the point) to do such transformation

One solution would be to project every lidar point into your image, then keep a track of which lidar point go in which image pixel. If you do so, you may implement the deth buffer (or z-buffer --> see on google for more detail) trick, in order to remove hidden points behind a closer object.

[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 25 March 2020 at 15:04
To: Kiarash Farivar <kiyarashfarivar@gmail.com>

Hi Kiarash,

I just recheck data, and I see that for the 3 "Jussy" dataset, there is a shift between Lidar and photogrammetric coordinate, you may use the other dataset, and I will inspect where this shift come from and how to correct it,

sorry for that,

Regards,

Adrien Gressin

On 25.03.20 11:54, Kiarash Farivar wrote:
[Quoted text hidden]

Kiarash Farivar <kiyarashfarivar@gmail.com> 25 March 2020 at 21:43
To: Adrien Gressin <adrien.gressin@heig-vd.ch>

Thank you for looking into that.

I have one more question, about the intensity values:
in the Leica las file intensities are all 0 except a few points that are 1. How should I interpret this ?
but in the photogrammetric file we have intensity values. Should I just use the intensities in the photogrammetric file?

[Quoted text hidden]

Kiarash Farivar <kiyarashfarivar@gmail.com> 27 March 2020 at 15:03
To: Salzmann Mathieu <mathieu.salzmann@epfl.ch>, Remelli Edoardo <edoardo.remelli@epfl.ch>

Hi
These are the emails I forgot to cc you on.

1.

Hi Kiarash,

You will find my answers below in your mail

Regards,

Adrien

On 25.03.20 11:54, Kiarash Farivar wrote:

> Thank you for the answers it cleared up some of my confusions. I have some more questions:
>
> 1. what application do you use for visualizing the .las files ?

You may use CloudCompare, which is free and opensource.

> 2. Where does the color information in the Leicia .las file come from? (is it from the images already provided ? or ...)
>
> about the transformations:

Leica color information come from internal sensor cameras, but we don't have access to such images (we may have such data using the C# API, but we need some dev to do it)

> 3. So the two .las files we have one is from the Leicia and the other is constructed from the images (using photogrammetry).
> How can I transform a location in the photogrammetry point cloud coordinates into a location in the Leicia point cloud coordinates?

All data are in the same coordinate system, so XYZ_photogrammetric = XYZ_leica

> 4. we need to know how each **pixel** of an image maps to a location in the **Leicia lidar** coordinates.
> How can I do this ?
> what you provided seems to be transformation from the **photogrammetry** coordinates (X,Y,Z) to the **image position** (x,y).
> also what does image position (x,y) mean? (what coordinates do we use ? where is the origin (0,0) ?)

For what I understand from the previous work (on CFF data), you may compute for each leica lidar point, the NN feature centered on the the corresponding pixel on the image, so for me you need the (X,Y,Z) to image positon (x, y), isn't it ?

If not, the other transformation is not possible for two reason :

- there is not a lidar point for each image pixel (you may have so hidden part)

- even if we suppose the latter, you may need to know the depth (distance from the camera to the point) to do such transformation

One solution would be to project every lidar point into your image, then keep a track of which lidar point go in which image pixel. If you do so, you may implement the deth buffer (or z-buffer --> see on google for more detail) trick, in order to remove hidden points behind a closer object.

2.

Hi Kiarash,

I just recheck data, and I see that for the 3 "Jussy" dataset, there is a shift between Lidar and photogrammetric coordinate, you may use the other dataset, and I will inspect where this shift come from and how to correct it,

sorry for that,

Regards,

Adrien Gressin

3.

[Quoted text hidden]

Kiarash Farivar <kiyarashfarivar@gmail.com> 27 March 2020 at 15:08
To: Adrien Gressin <adrien.gressin@heig-vd.ch>
Cc: Salzmann Mathieu <mathieu.salzmann@epfl.ch>, Remelli Edoardo <edoardo.remelli@epfl.ch>

Hello Adrien
sorry to bother you again but I have another question regarding the process of photogrammetry. How do you perform that ? What is the algorithm used ?
[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 30 March 2020 at 18:16
To: Kiarash Farivar <kiyarashfarivar@gmail.com>
Cc: Salzmann Mathieu <mathieu.salzmann@epfl.ch>, Remelli Edoardo <edoardo.remelli@epfl.ch>

Helo Kiarash,

Regarding the intensity, it's true and I would have imagined the opposite...

I think the intensity of the photogrammetric point-cloud just comes from the gray level derivated from RGB color. About the RTC data, there must have been a format conversion issue, with a threshold, because the white dots are the most reflective parts of the area.

Do you use such information for your classification ? If so, I can see if we can retrieve that information.

About the photogrammetric process, we use the well-known "structure from motion" method : https://en.wikipedia.org/wiki/Structure_from_Motion

A lot of implementation / software exists such as proprietary software :

- Agiosft Metashape (the one we use here)

- Pix4D (a spin-off of EPFL)

You may find a lot of open-source solution as well :

- Micmac (French solution)

- OpenSfM

- Bundler / PMVS

Regards,

Adrien

[Quoted text hidden]

Kiarash Farivar <kiyarashfarivar@gmail.com> 3 April 2020 at 11:40
To: Adrien Gressin <adrien.gressin@heig-vd.ch>

Hello

> Regarding the intensity, it's true and I would have imagined the opposite...
>
> I think the intensity of the photogrammetric point-cloud just comes from the gray level derivated from RGB color. About the RTC data, there must have been a format conversion issue, with a threshold, because the white dots are the most reflective parts of the area.
>
> Do you use such information for your classification ? If so, I can see if we can retrieve that information.

The paper I use as a reference uses the intensity of the points as a feature to train the neural network so it would be great if we could get that.
Thanks
[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 3 April 2020 at 14:16
To: Kiarash Farivar <kiyarashfarivar@gmail.com>

Hi,

I will try to get back this information for you, but it'll take me some times because we now have one pointcloud with classification data and another one with intensity data, and we will have to merge such information.

An other solution would be to use color data as intensity data, since lidar beam is in near-infrared channel, if you use red color data you would have similar information.

Regards,

Adrien

[Quoted text hidden]

Kiarash Farivar <kiyarashfarivar@gmail.com> 3 April 2020 at 15:54
To: Adrien Gressin <adrien.gressin@heig-vd.ch>

Thank you.
[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 23 April 2020 at 11:33
To: Kiarash Farivar <kiyarashfarivar@gmail.com>

Hi Kiarash,

Please find the new lidar dataset, with intensity, classification and correct georeferencing here :

https://filesender.switch.ch/filesender/?vid=11f783df-f440-1229-7311-000015a5c066

Hope you will soon have firsts results to show us,

Regards,

Adrien

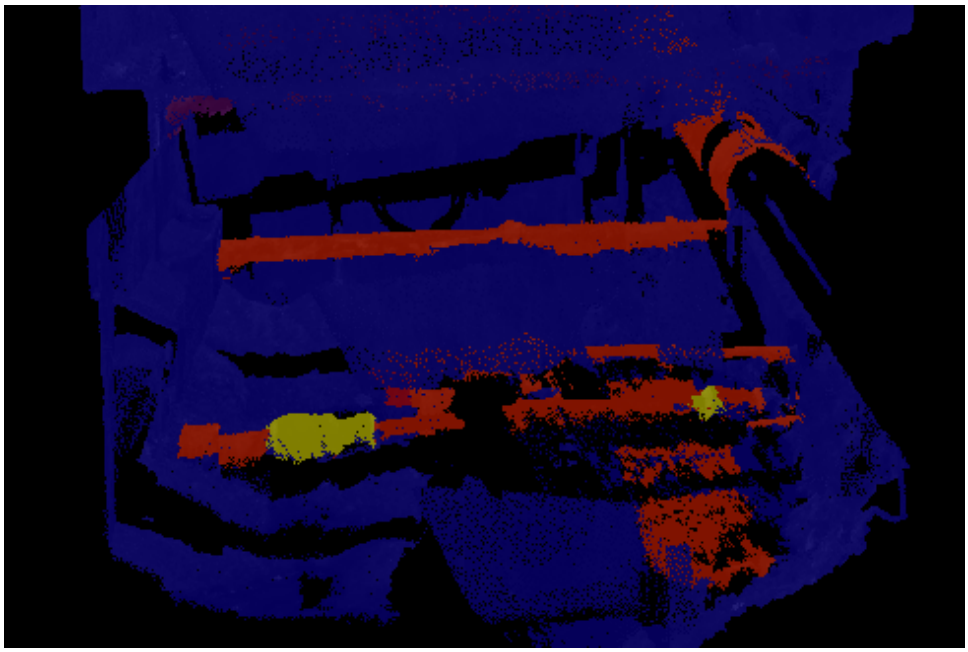
[Quoted text hidden]

Kiarash Farivar <kiyarashfarivar@gmail.com> 24 April 2020 at 00:39
To: Adrien Gressin <adrien.gressin@heig-vd.ch>

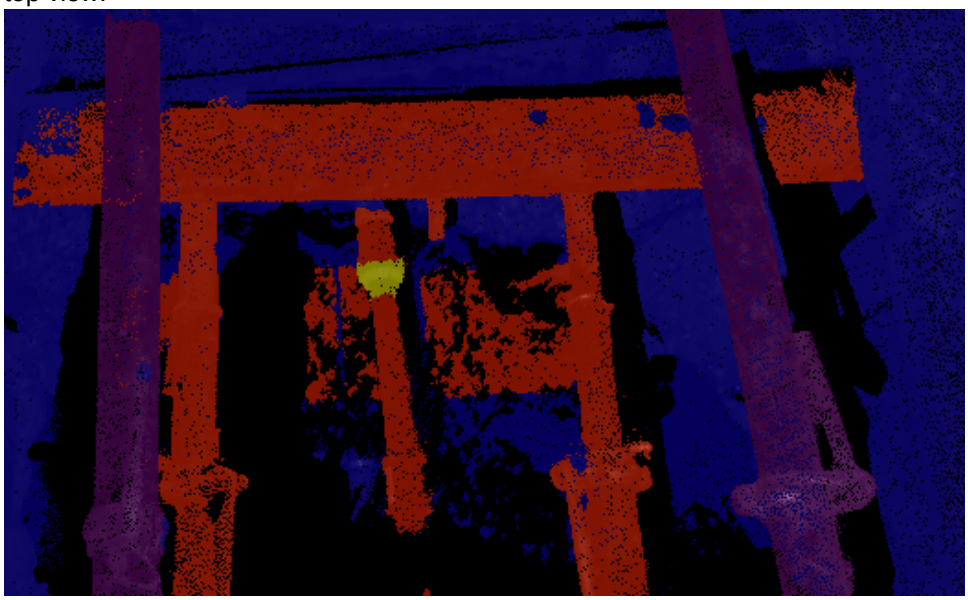
Thank you,

1. would you explain what has changed in the new lidar data ? has the coordinates of the points changed ?

2. I also found this one weird classification in red here (class 3) in jussy1. It looks like it is part of the ground. was this intentional ? (should I include it ?)
it is in the very bottom of the scene.
side view:



top view:



3. In the folder jussy2; I was trying to use the functions you provided for transforming from world coordinates into image coordinates. But the calibration file seems to be missing some of the values.

```
<?xml version="1.0" encoding="UTF-8"?>
<calibration>
  <projection>frame</projection>
  <width>9504</width>
  <height>6336</height>
  <f>6577.66705936027</f>
  <cx>-23.2465197793531</cx>
  <cy>-26.180852621249</cy>
  <k1>-0.134754632157509</k1>
  <k2>0.106619395464137</k2>
  <k3>-0.00936108294659722</k3>
  <p1>-0.00029845208467988</p1>
  <p2>-0.000847821870749913</p2>
  <date>2020-02-17T14:24:53Z</date>
```

</calibration>

4. In the pictet folder sony photos; I guess the calibration is not correct because when I transformed the points all of them where mapped to the same point. (the min and max of x and y were the same) and they didn't fall into the width and height of the corresponding photo. (I tried it for photo 'DSC00947') . Also the for the iphone_video images there was no calibration or opk files. it would be good to have them.

thank you
[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 12 May 2020 at 12:49
To: Kiarash Farivar <kiyarashfarivar@gmail.com>

Hi Kiarash,

I hope your work is going well, may we have some news about your firsts results ?

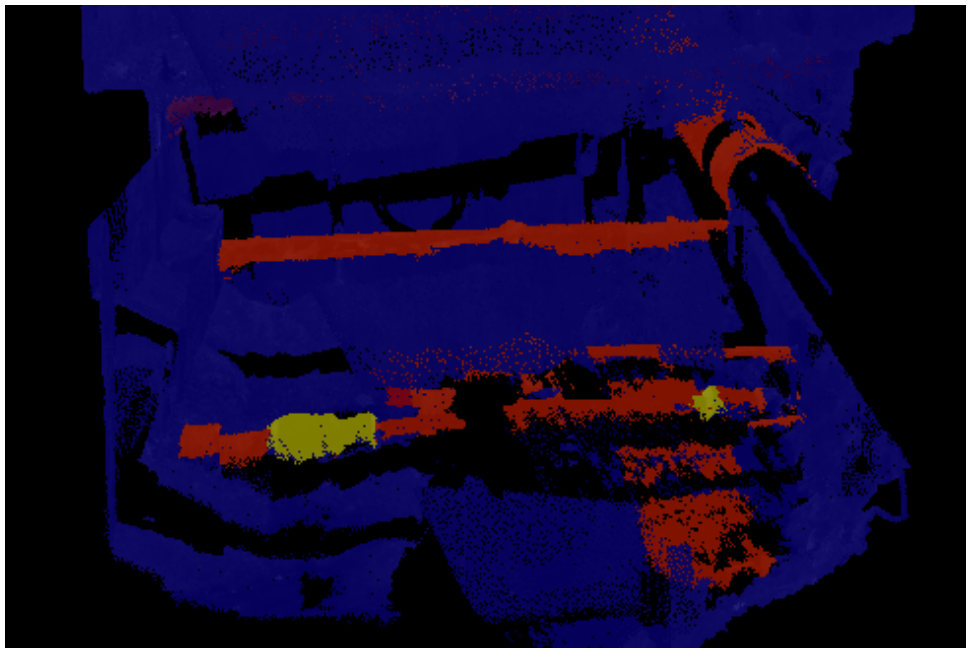
Here are my answer about your last question,

On 24.04.20 00:39, Kiarash Farivar wrote:

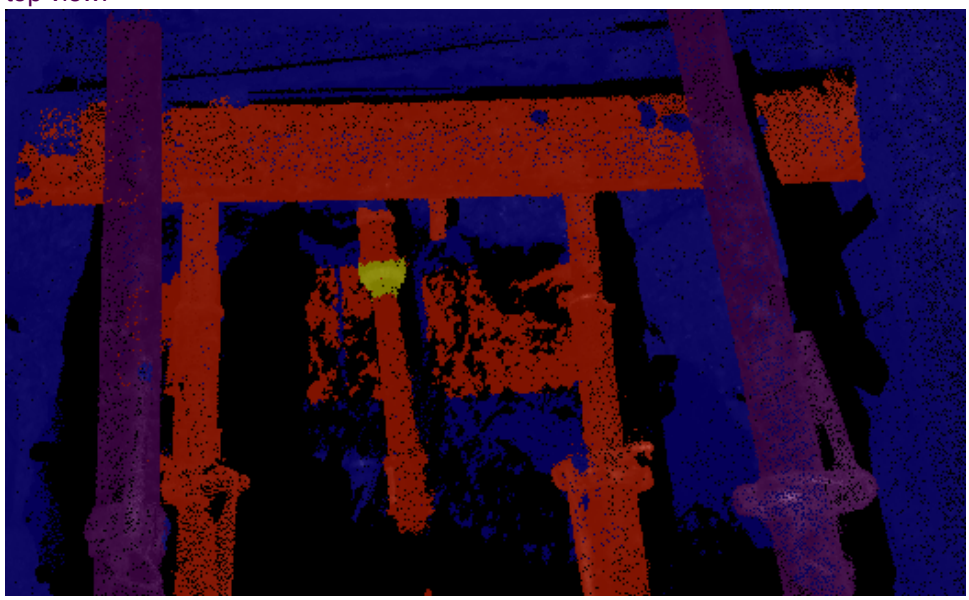
> Thank you,
>
> 1. would you explain what has changed in the new lidar data ? has the coordinates of the points changed ?

As you ask for, the new LiDAR include intensity data.

> 2. I also found this one weird classification in red here (class 3) in jussy1. It looks like it is part of the ground. was this intentional ? (should I include it ?)
> it is in the very bottom of the scene.
> side view:



top view:



It's clearly a classification mistake I made, sorry for that.

3. In the folder jussy2; I was trying to use the functions you provided for transforming from world coordinates into image coordinates. But the calibration file seems to be missing some of the values.

```
<?xml version="1.0" encoding="UTF-8"?>
<calibration>
  <projection>frame</projection>
  <width>9504</width>
  <height>6336</height>
  <f>6577.66705936027</f>
  <cx>-23.2465197793531</cx>
  <cy>-26.180852621249</cy>
```

> <k1>-0.134754632157509</k1>
> <k2>0.106619395464137</k2>
> <k3>-0.00936108294659722</k3>
> <p1>-0.00029845208467988</p1>
> <p2>-0.000847821870749913</p2>
> <date>2020-02-17T14:24:53Z</date>
> </calibration>

Missing values must be fixed to 0.0 (I think you are speaking about k4, c1 and c2 ?).

> 4. In the pictet folder sony photos; I guess the calibration is not correct because when I transformed the points all of them where mapped to the same point. (the min and max of x and y were the same) and they didn't fall into the width and height of the corresponding photo. (I tried it for photo 'DSC00947') . Also the for the iphone_video images there was no calibration or opk files. it would be good to have them.

The issue come from the LAS file which have a bad georeferencing metadata, I just send you the correct one with Switchfilesender.

The iphone_video data of Pictet area doesn't include any new pipes, since it only concern the underground part, whereas data from the RTC and the sony camera are on the outside part (with the new pipes not yet installed). Thus, we do not provide OPK and calibration data here.

Regards,

Adrien

[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 14 May 2020 at 14:00
To: Kiarash Farivar <kiyarashfarivar@gmail.com>

Hi Kiarash,

I just received a "delivery delay issue" for my last email, do you already receive it ?

Regards,

Adrien

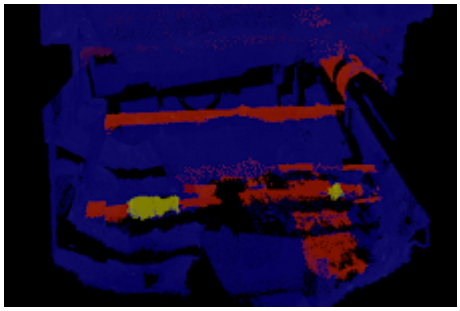
[Quoted text hidden]

Kiarash Farivar <kiyarashfarivar@gmail.com> 15 May 2020 at 15:03
To: Adrien Gressin <adrien.gressin@heig-vd.ch>

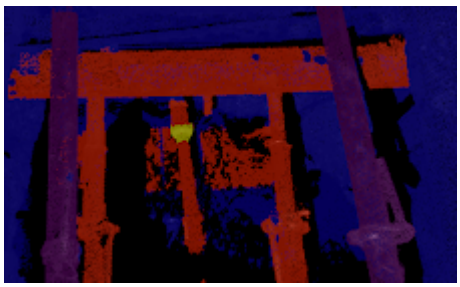
Hello
I received your previous email May 12th.

Thank you for the corrections.
Regarding the results my work is still in progress. I will write a report by June 5th .
[Quoted text hidden]

2 attachments



jussy1_problem1.png
84K



jussy1_2.png
143K

Kiarash Farivar <kiyarashfarivar@gmail.com> 25 May 2020 at 16:19
To: Adrien Gressin <adrien.gressin@heig-vd.ch>

Hello
I have a question about the opk files used in translating points from lidar to image coordinates.
I think the Omega, Phi, Kappa angle values are used to calculate the rotation matrix (r11, r12, r13, r21, r22, r23, r31, r32, r33).
What is the formula used to get the rotation matrix from the angles ?
thank you
[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 25 May 2020 at 16:47
To: Kiarash Farivar <kiyarashfarivar@gmail.com>

Hello,

Hi Kiarash,

You will find some code to convert O,P,K angular to rotation matrix at the end of my mail.

you can find more information here :

https://support.pix4d.com/hc/en-us/articles/202558969-Yaw-Pitch-Roll-and-Omega-Phi-Kappa-angles#Omega,%20Phi,%20Kappa%20definition

or here :

https://en.wikipedia.org/wiki/Rotation_formalisms_in_three_dimensions#Euler_angles_(z-y%E2%80%B2-x%E2%80%B3_intrinsic)_%E2%86%92_rotation_matrix

Regards,

Adrien

```
import numpy as np
import math

def rot_x(a):
    R = np.array([
        [1,0 ,0 ],
        [0, np.cos(a),np.sin(a)],
        [0,-np.sin(a),np.cos(a)]
    ])
    return R

def rot_y(a):
    R = np.array([
        [np.cos(a),0 ,-np.sin(a)],
        [0,1,0],
        [np.sin(a),0,np.cos(a)]
    ])
    return R

def rot_z(a):
    R = np.array([
        [ np.cos(a),np.sin(a),0 ],
        [-np.sin(a),np.cos(a),0],
        [0,0,1]
    ])
    return R

def rot_opk(o,p,k):
    Rx = rot_x(o)
    Ry = rot_y(p)
    Rz = rot_z(k)
    return Rz.dot(Ry).dot(Rx)

def rot_opk_degrees(o_deg,p_deg,k_deg):
    o_rad = np.radians(o_deg)
    p_rad = np.radians(p_deg)
    k_rad = np.radians(k_deg)
    return rot_opk(o_rad,p_rad,k_rad)
```

[Quoted text hidden]

Kiarash Farivar <kiyarashfarivar@gmail.com> 25 May 2020 at 21:01
To: Adrien Gressin <adrien.gressin@heig-vd.ch>

thank you very much for the quick answer.
I just want to check two more things.
1. in the rotation and translation function "colinearite" R and S seem to be independent. am I correct to assume that ?
2. what is the role of the "correct_disto_agisoft" function when doing the transformation ?

[Quoted text hidden]

Adrien Gressin <adrien.gressin@heig-vd.ch> 25 May 2020 at 21:51
To: Kiarash Farivar <kiyarashfarivar@gmail.com>

Hi again,

On 25.05.20 21:01, Kiarash Farivar wrote:

> thank you very much for the quick answer.
> I just want to check two more things.
> 1. in the rotation and translation function "colinearite" R and S seem to be independent. am I correct to assume that ?

I don't know what do you mean by independent, but there is no direct link between thus two data, first R is the orientation of the camera (in which direction the camera is looking at, and S is the position of the camera).

> 2. what is the role of the "correct_disto_agisoft" function when doing the transformation ?

The colinearity function allow to transform terrestrial coordinate (XYZ) to image coordinate (i,j) knowing the camera position, orientation and the focal, with the simplified "Pinhole camera model". But due to optical deviation in the camera, a distortion model must be fitted (in a calibration sted) and used to correct the image coordinate. Very often, distortion is corrected with a Brown-Conrady model (https://en.wikipedia.org/wiki/Distortion_(optics)), this model is used in Agisoft, and this is what the "correct_disto_agisoft" does.

To use this latter, you must read distortion parameter from the calibration file (.cam) of each camera (available with eahc opk file).

Regards,

Adrien Gressin

[Quoted text hidden]