TSBB09 Laboratory Assignment: Range camera

Developed 2008 by Maria Magnusson and Fredrik Viksten.
Computer Vision Laboratory, Linköping University, Sweden

Last update: December 2020

Contents

1 Preparations

Before coming to the computer session it is necessary to rehearse the course material on range cameras. The most important for this laboratory assignment is the lecture slides that refer to [1].

You will also find a number of home exercises to be answered before the session. They are all clearly marked with a pointing finger.

Extra

Implementation exercises marked with the **EXTRA** symbol may be temporarily skipped and completed when the other exercises are finished.

2 Introduction

Note that there is a *utility box* with different things (calibration object, white wall socket, red house, etc.) necessary for the lab. In the video `UtilityBox.mp4`, the objects within the box are inverstigated. In the case of range camera images, there are four different types of images:

- The *intensity image* $f(s,t)$, where f is the intensity on the sensor, shows a weak image of the object (if we have ambient background light) and a curved laser line which shape depends on the height profile of the object, see Fig. 1.6 in [1].
- A *range image* $r(x,y)$, where r is the distance to the object and (x,y) is the spatial coordinates, is shown to the right in Fig. 1.1 in [1]. (The image to the left is a normal intensity image.)
- In a *pseudo range image* $s_L(x,t)$, s_L is the coordinate of the laser line position, x is the coordinate of the laser sheet and t is a sensor coordinate. The pseudo range image looks rather similar to the real range image $r(x,y)$. In some cases, they are just scaled linearly in relation to each other, see for example the arrangements in Fig. 2.4 (2.5) and 2.6. Then r is proportional to s (Eq. (2.16)) and y is proportional to t (Eq. (2.24) with $\alpha = 0$). The arrangement in Fig. 2.7 does not apply to a linear scaling relationship, but the pseudo range image and the correct range image still looks rather similar.
- In the *pseudo intensity image* $f_{sL}(x,t)$, f_{sL} is the intensity of the reflected laser line. Consequently, the pseudo range image and the pseudo intensity image are closely related to each other. Note that the pseudo intensity image is not a normal intensity image. It shows the reflectance of the object, but the geometry is somewhat unusual.

3 Theory questions about the range camera lab setup

This lab setup is demonstrated in the video `RangeCameraLabSetup.mp4`. Look at the video and answer the following questions.

QUESTION 1: Which figure among Fig. 2.4-2.8 in [1] corresponds to the range camera lab setup?

QUESTION 2: Give an approximate value for α of the range camera lab setup.

QUESTION 3: Give an approximate value for the desired β by looking in table 3. Suppose that $18\text{mm} \leq f \leq 75\text{mm}$.

QUESTION 4: The range camera lab setup has a sensor that is perpendicular to the optical axis, ie. $\beta = 0$. What is the consequence when β not equals the desired value?

QUESTION 5: How serious is the β discrepancy in our case?

QUESTION 6: Which equations are valid for r and y in the range camera lab setup? (Give only the equation numbers.)

---

4 Basic images from the range camera lab setup

We will now work with images from the range camera.
The image files are at `/courses/TSBB09/RangeCamera` and in Lisam.
Look at the video and answer the following questions.

First open MATLAB and set up the path:

```
>> fpath='/courses/TSBB09/RangeCamera/';
```

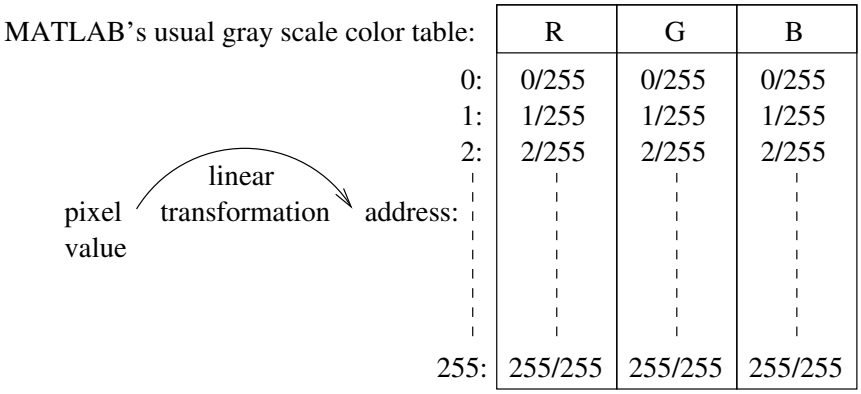
We will start with the pseudo range image `handran`. Give the following commands:

```
>> fp = fopen([fpath 'handran'],'r');
>> a = fscanf(fp,'%d',[500,256]); % image size:  500*256
>> figure(1)
>> imagesc(a);
>> axis image; colorbar;
>> colormap([(0:255)'/255,(0:255)'/255,(0:255)'/255])
```

The last command is equivalent with

```
>> colormap gray(256)
```

This color map is illustrated below.



The hand looks a bit dark. Make it brighter with:

```
>> imagesc(a,[0,100]);
```

Have you noticed the strange high values close to the finger edges? Load the pseudo intensity image `handint` to get an explanation:

```
>> fp = fopen([fpath 'handint'],'r');
>> b = fscanf(fp,'%d',[500,256]); % image size:  500*256
>> figure(2)
>> imagesc(b);
>> axis image; colorbar;
>> colormap([(0:255)'/255,(0:255)'/255,(0:255)'/255])
```

Note that the strange values in the pseudo range image correspond to low values in the pseudo intensity image. The reasons for low values in the pseudo intensity image can be:

- The object is dark and has low reflectivity.

- The object is shiny and gives specular reflection in a hidden direction for the camera.
- There is *laser occlusion*, i.e. the laser has not been able to illuminate an object point visible from the camera.
- There is *camera occlusion*, i.e. the camera can not see the object point that the laser illuminates.

Let us combine the pseudo range image with the pseudo intensity image so that the strange values disappear (are set to 0).

```
>> figure(3);
>> c = a.*(b>T);
>> imagesc(c);
>> axis image; colorbar;
>> colormap gray
```

QUESTION 7: Determine a suitable value for `T` above!

Let us use another, more colorful, colortable.

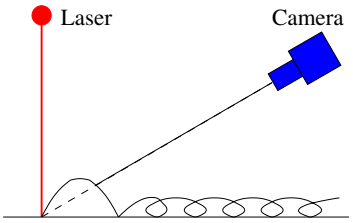
```
>> rainbow = jet;
>> colormap(rainbow)
```

QUESTION 8: The values that we set to 0 before should rather be marked as uncertain. Perform this by changing the colortable so that the uncertain values are shown in gray. What commands do you give?

The effect of highlighting uncertain range values are actually even more evident for `musran`, a pseudo-range image of a computer mouse. On this image, the uncertain values have already been set to 0.

```
>> fp=fopen([fpath 'musran'],'r');
>> b = fscanf(fp,'%d',[250,256]); % image size:  250*256
>> imagesc(b);
```

The situation when this image was recorded was probably as illustrated in the image below.



QUESTION 9: What type of occlusion is this?

We will now look at an intensity image $f(s,t)$ and try to detect the laser line.

```
>> fp = fopen([fpath 'wood'],'r');
>> g = fscanf(fp,'%d',[512,512]); % image size:  512*512
>> imagesc(g);
```

QUESTION 10: The simplest method is to look for the maximum value along every row/column of the image. Your goal is to plot the profile $S_L(x = const, t)$. Use the MATLAB-commands `[maxint, maxindex] = max(x,[],dim)` and `plot`. Give your MATLAB-commands and sketch the profile below.

QUESTION 11: How can you locate the uncertain values?

QUESTION 12: By using the information in `maxint`, tidy up the profile by setting some of the values to uncertain, `NaN` (not a number). Give your commands below:

---

5 Calibration of the range camera lab setup

QUESTION 13: Rehearse the exercise regarding calibration of a flat world, a homography, in the laboratory assignment regarding camera calibration. At the computer exercise you will do a similar calibration task, but then the flat world will be the laser sheet. How many points on a real calibration object inserted in the laser sheet (r_i, y_i) is nescessary to measure?

QUESTION 14: For your help there is a non-completed MATLAB code in `/courses/TSBB09/RangeCamera/calibration.m`. See also the MATLAB section in the end this laboratory assignment. Make a sketch of the calibration object in the *utility box* and the correponding calibration points $(Y1, R1), (Y2, R2), ..., (Y8, R8)$ in `calibration.m`.

QUESTION 15: Measure corresponding points in the calibration image `calim`.

QUESTION 16: Compute and give the calibration matrix `C`.

QUESTION 17: Check your C-matrix by transforming $(Y1, R1) = (0, 17)$.

QUESTION 18: The test object is a box with a parallel trapezoidal cross-section. Your task is to measure the circumference of it. Use the image `trapez` and maybe also the image `notrapez`. Compare your computed value with a measure of the real test object. You should get a good agreement with error $< 5\%$.

---

6 Measures with an industrial range camera

In this part of the lab we are using a Ruler E600 from SICKIVP. It is built on the same basic techniques that you have worked with in the previous parts of the lab but there are some differences. First and foremost, the Ruler range of cameras comes with a fixed geometry and are fully calibrated. This means that your measurement values are given in millimeter. Before we begin with the exercises there are some things that you need to know about the Ruler camera. Images and information in this section are taken from [2]. A demonstration is in the video `RangeCameraIndSetup.mp4`.

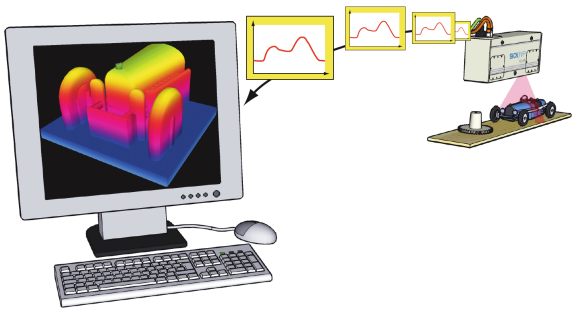
6.1 Laser security

This product is equipped with a class 2M laser. DO NOT STARE DIRECTLY INTO THE LASER BEAM/SHEET! This is very important since you might loose or damage your eye sight if you do. The Ruler we are using in this lab is mounted so that the laser beam comes out of the side of the camera (it is more common to mount it so that the laser points down) so please take extra caution in not looking into the laser. There is however no danger in being in the same room as the camera while the laser is operating as long as you do not stare directly into the laser.

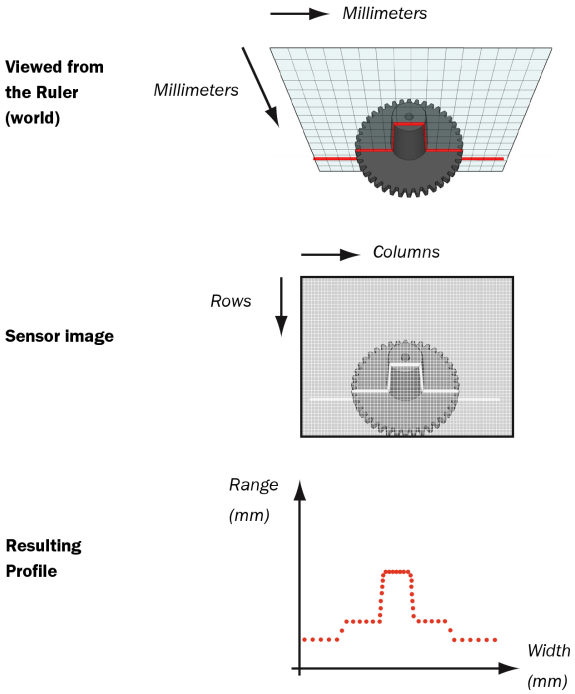
6.2 Outputs of the Ruler E600

The range cameras from SICKIVP all work as a high-speed source for height profiles. The height profiles are sent to a computer workstation via gigabit Ethernet in the sequence that they are captured. The capturing of anything

else than a static height profile is dependent on some translation, either the objects are translated (most common usage) or the camera itself is translated. A range "image" is built up from the output profiles by concatenating them side by side, see the figure below.

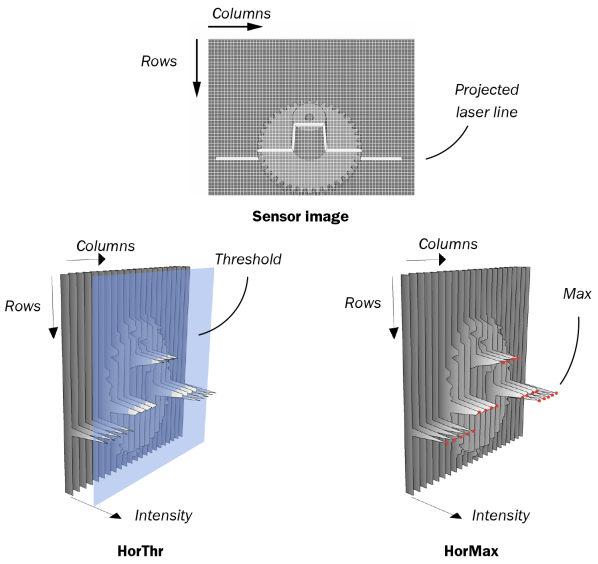


Besides the height profile it also sends an intensity profile which again can be used to build an intensity image of the scene. The intensity image has, as was mentioned in the introduction, a somewhat unusual geometry and can not be treated as an ordinary image. Since the calculation of the height profile is done in the camera (or rather on the chip) the camera can operate at a very high speed without oversaturating the Ethernet link. The camera has a max capacity of outputting roughly 10000 profiles / second. The following image illustrates what the camera does.

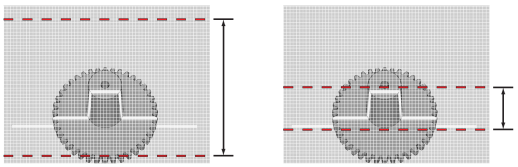


6.3 Measurement methods

From the previous section it should come as no surprise that the accuracy in height estimation is dependent on the accuracy with which the peak of the laser line can be detected. There are of course a number of ways for finding the peak, all with different processing characteristics. For this reason there are a number of different methods used to find the laser line in each sensor column with either speed or accuracy. Here is an illustration of two of them.

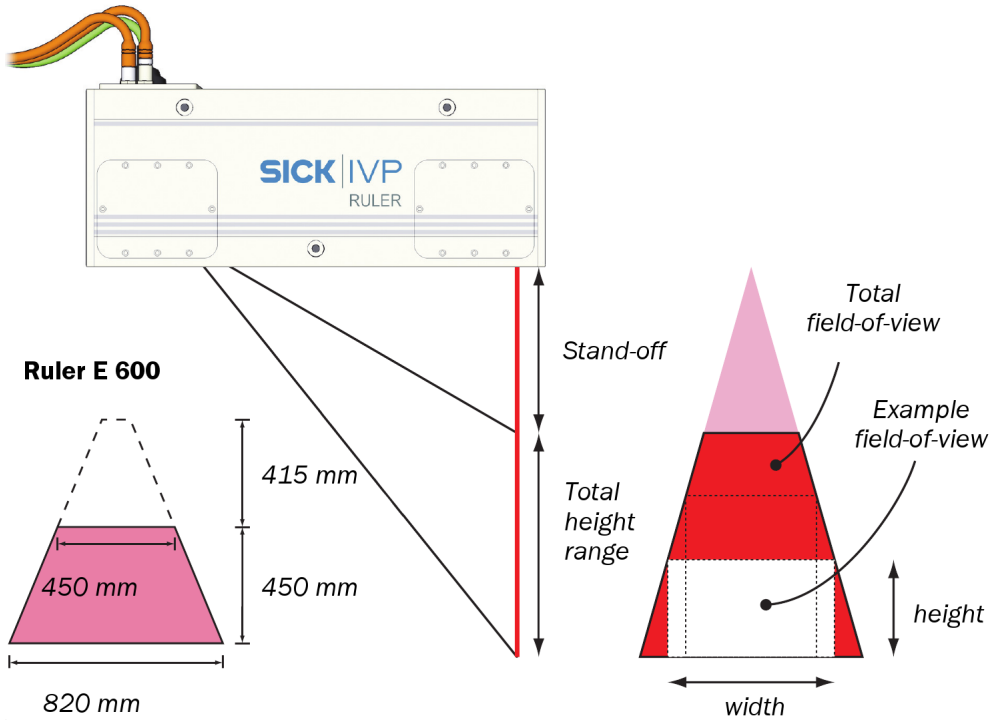


The methods called **HorThr** (short for *Horizontal Threshold*) and **HorMax** (short for *Horizontal Maximum*) are the fastest methods. There is also one called **HorMaxThr** and one called **Hi3D**. The last two do find the laser line down to 1/2 pixel and 1/16 pixel respectively. Hi3D is the slowest of all the methods available. The methods all operate in parallel for each sensor column so there is only one way to increase the speed of the measurement. Depending on the geometry and the shape and size of the objects that are measured it might be possible to decrease the number of sensor rows that each method operates on.



6.4 Geometry and measurement volume restrictions

Due to the geometry of the camera and the field-of-view of the inbuilt 2D camera there are naturally some restrictions on the measurement volume. The stand-off is the cloesest distance any measurement can be made at, the total height range is only available if all sensor rows are utilized (see previous section) and the possible width varies with the choosen height.



QUESTION 19: From the figure above we can see that each range "pixel" will cover a different area on the object dependent on the distance to the sensor. Based on this information, where in the volume will we have the most accurate measurements?

6.5 Measuring tiny details

In the file "cardim" there is a partial scan of a credit card. Your job is to read the numbers and any text that you can from the scan. This does not have to be automatic but you should convince the laboration assitant that you've as good a view of the text as you could get. Load the scan:

```
>> load cardim;
```

Each time you load a range image like that you get a variable called "rangeim". Type

```
>> rangeim
```

to see the different parts of the range scan that are available.

Make a range data loading MATLAB-script of the following code:

```
>> tick_divide = 1;
>> r = rangeim.range;
>> y = rangeim.x;
>> x = repmat(double(rangeim.prof_id)/tick_divide,[size(y,1)
1]);
>> imsz = size(r);
>> R1 = zeros([imsz 3]);
>> R1(:,:,1) = double(y);
>> R1(:,:,2) = double(x);
>> R1(:,:,3) = double(r);
>> cert = (r == 0);
>> R1c = repmat(cert,[1 1 3]);
>> R1(R1c) = NaN;
```

Later on you will see that the mesh command will not draw anything if a point is NaN (not a number).

QUESTION 20: Regard the following row,

```
>> x = repmat(double(rangeim.prof_id)/tick_divide,...
```

where `rangeim.prof_id` is a 1D integer linear scale and `tick_divide` is a scaling factor. Print `x` in the MATLAB window and check if it is a linear scale in the row-direction, column-direction or both!

QUESTION 21: Why is it appropriate with a linear scale for `x`?

When your MATLAB-script is working it should output a datastructure `R1(:,:,:)` of size $N \times M \times 3$, i.e. $N \times M$ 3D-points (x, y, r).

QUESTION 22: Load your range data into the variable R1 and try both:

```
>> mesh(R1(:,:,3)); axis equal
>> mesh(R1(:,:,1),R1(:,:,2),R1(:,:,3)); axis equal
```

Which one is theoretically the best?

QUESTION 23: Adjust `tick_divide` so that you can read parts of the text. Try e.g. 1, 2, 3, 4, 5. What is a good value for `tick_divide`? Write down parts of the text!

QUESTION 24: Why do you need to adjust `tick_divide` if the camera is calibrated?

Now load the next piece of data:

```
>> load letter.mat;
```

QUESTION 25: Which letter is found in the range scan?

QUESTION 26: Look in the *utility box* for the object that was input to the ruler. Do you agree that there is absolutely no height difference at all between the letter and its background?

QUESTION 27: Explain why there is a height difference!

6.6 Volume measurement of object

First we will lock at a scan of a white wall socket. Load the scan:

```
>> load vit_vaggkontakt;
```

and arrange the range data in `R1(:,:,:)` by using the MATLAB-script you wrote in the previous subsection.
Then crop and show the data according to:

```
>> R2 = R1(401:700,201:700,:);
>> Intens = rangeim.intens(401:700,201:700);
>> figure(1)
>> meshc(R2(:,:,1),R2(:,:,2),R2(:,:,3))
>> figure(2)
>> subplot(2,2,1), imagesc(R2(:,:,1)), title('y image');
>> subplot(2,2,2), imagesc(R2(:,:,2)), title('x image');
>> subplot(2,2,3), imagesc(R2(:,:,3)), title('r image');
>> subplot(2,2,4), imagesc(Intens(:,:)), title('i image');
>> figure(3)
>> subplot(2,2,1), plot(R2(:,320,1)), title('vert, y');
>> subplot(2,2,2), plot(R2(150,:,2)), title('horiz, x');
>> subplot(2,2,3), plot(R2(:,320,3)), title('vert, r');
>> subplot(2,2,4), plot(R2(150,:,3)), title('horiz, r');
```

Note that you now also show the intensity image that corresponds to the range data. Also look at the real wall socket in the *utility box*. In the data there are some areas marked with NaN as before, but there are also some distortions.

QUESTION 28: Where are the distortions and what is the reason for them?

QUESTION 29: How can the distortions be detected and "removed" (marked with NaN)?

Due to the distortions, it will not be possible to measure the volume of the wall socket exactly. It is, however, not always nescessary to get an exact volume measure and sometimes the result can be improved by using a' priori knowledge.

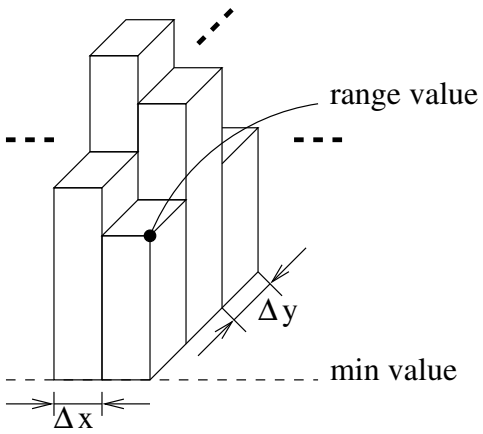
We will now use the wall socket to determine a better value for `tick_divide`. In the following two tasks, use the real wall socket and the MATLAB figures (2) and (3) from the code above.

QUESTION 30: What is the width of the real wall socket?

QUESTION 31: What is the width of the wall socket in the y-direction in mm according to the MATLAB figures (2) and/or (3)? (You need to click in the figures and check the values.) Does the measurement agree with the real measure?

QUESTION 32: What is the width of the wall socket in the x-direction in pixels according to the MATLAB figures (2) and/or (3)? (You need to click in the figures and check the values.) Use this information to determine a new value for `tick_divide`! **This value is important since you will need it in the next task. Let the teacher check the value!**

Your task is now to write a MATLAB program that measures the volume of a small red "house" made of wooden blocks. The real house is in the *utility box*. There are four scans with different rotation angles in `hus0`, `hus90`, `hus_lutande0` and `hus_lutande90`. Get inspiration from the Riemann-integral and the figure below showing volume elements.



QUESTION 33: The value for Δx is the same for all volume elements, but the value for Δy varies. Why?
Hint: Check the last figure in Section 6.2.

To calculate the volume, start by cropping and showing the data as you did for the white wall socket. The MATLAB commands `diff` and `nansum` will be useful.

QUESTION 34: Which volume do you get for `hus0`?

QUESTION 35: *Check 1:* Take the other scan, `hus90`, and check so that you get approximately the same volume. Which volume do you get now?

QUESTION 36: *Check 2:* Measure the volume of the real red house with a ruler!

6.7 Fit a plane to points

Extra

The equation for a plane is $x_1p_1 + x_2p_2 + x_3p_3 - L = 0$, where (p_1, p_2, p_3) is the normal to the plane and (x_1, x_2, x_3) is a point in the plane. A simple solution is to form two vectors from three points in the plane and compute the cross product. A more careful solution is the following. The equation

$$\begin{pmatrix} x_1 & x_2 & x_3 & 1 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ -L \end{pmatrix} = 0$$

is the plane equation. Filling with more points gives

$$\begin{pmatrix} x_1 & x_2 & x_3 & 1 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ -L \end{pmatrix} = 0,$$

i.e. $\mathbf{Xp} = 0$. This homogeneous equation system can be solved by `svd`:

```
>> [U,D,V]=svd(X);
```

where V is a 4×4-matrix. The vector corresponding to the smallest singular value, $V(:,4) = (a,b,c,d)^T$, gives the best solution to $\mathbf{p}$ i.e.

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ -L \end{pmatrix} \approx k \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}.$$

Extra

QUESTION 37: It was rather difficult to read all the text on the card previously, right? Now when you have learned to adjust a plane, you can use that on the card image. Subtract the plane from the range and show the result using `imagesc`. What is the whole text on the card?

Extra

QUESTION 38: To measure the volume of `hus_lutande0` and `hus_lutande90`. you need to fit a tilted plane to the bottom. Which volumes do you get?

References

[1] Mattias Johannesson, "Active Range Imaging 2" from "SIMD Architectures for Range and Radar Imaging", *Ph. D. thesis No. 399*, Department of Electrical Engineering, Linköping University, Linköping, Sweden, 1995

[2] SICKIVP, "Ruler E operating instructions", *Document number 8011483*, SICKIVP, 2005-11-22

7 MATLAB files

7.1 calibration.m

```
% Calibration points in the laser plane
%--------------------------------------
Y1 =   0;   R1 =  17;
Y2 =  36.5; R2 =  36.5;
Y3 =  53.5; R3 =  73;
Y4 =  70.5; R4 = 109.5;
Y5 =  87.5; R5 = 109.5;
Y6 = 104.5; R6 =  73;
Y7 = 121.5; R7 =  36.5;
Y8 = 158;   R8 =  17;

% Calibration points in the image
%--------------------------------
u1 = ; v1 = ;
u2 = ; v2 = ;
u3 = ; v3 = ;
u4 = ; v4 = ;
u5 = ; v5 = ;
u6 = ; v6 = ;
u7 = ; v7 = ;
u8 = ; v8 = ;

f = [u1 v1 u2 v2 u3 v3 u4 v4 u5 v5 u6 v6 u7 v7 u8 v8]';

% Calibration matrix
%-------------------
D = [
    Y1 R1  1   0  0  0 -u1*Y1 -u1*R1;
     0  0  0  Y1 R1  1 -v1*Y1 -v1*R1;
    Y2 R2  1   0  0  0 -u2*Y2 -u2*R2;
     0  0  0  Y2 R2  1 -v2*Y2 -v2*R2;
    Y3 R3  1   0  0  0 -u3*Y3 -u3*R3;
     0  0  0  Y3 R3  1 -v3*Y3 -v3*R3;
    Y4 R4  1   0  0  0 -u4*Y4 -u4*R4;
     0  0  0  Y4 R4  1 -v4*Y4 -v4*R4;
    Y5 R5  1   0  0  0 -u5*Y5 -u5*R5;
     0  0  0  Y5 R5  1 -v5*Y5 -v5*R5;
    Y6 R6  1   0  0  0 -u6*Y6 -u6*R6;
     0  0  0  Y6 R6  1 -v6*Y6 -v6*R6;
    Y7 R7  1   0  0  0 -u7*Y7 -u7*R7;
     0  0  0  Y7 R7  1 -v7*Y7 -v7*R7;
    Y8 R8  1   0  0  0 -u8*Y8 -u8*R8;
     0  0  0  Y8 R8  1 -v8*Y8 -v8*R8];

% Calculate C via the pseudo inverse
%-----------------------------------

%...

% Check if the point (0,17) is correctly transformed
%---------------------------------------------------
test = C * [0 17 1]';
test(1) = test(1)/test(3);
test(2) = test(2)/test(3)
```