Image processing for extracting features

Clément Chenevas-Paule
Alexandre Audibert
Daria Senshina
Roman Kozhevnykov

Rockfalls

Figure 1

Figure 2

Experimental setup


Figure 3


Figure 4

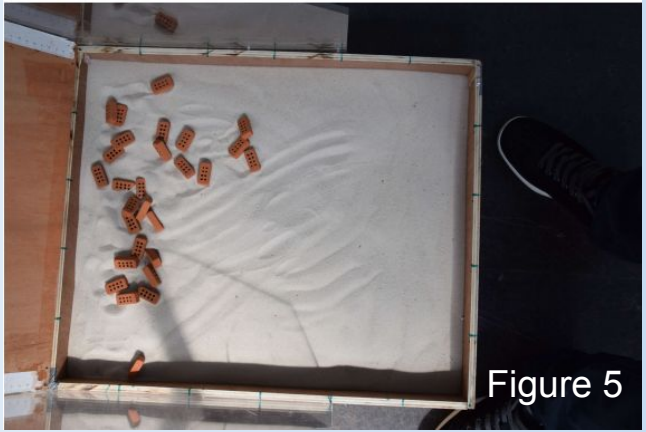


Figure 5

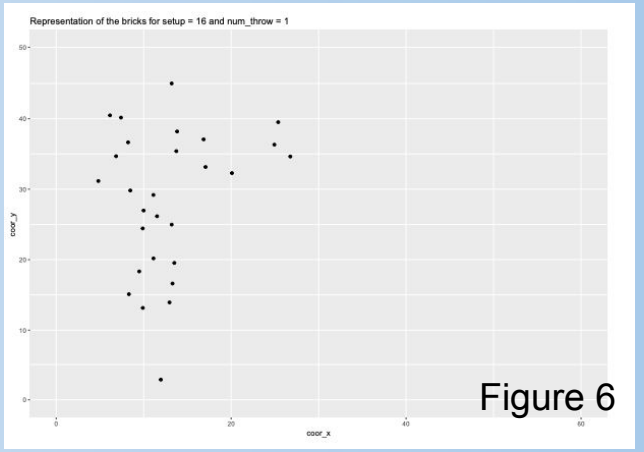


Figure 6

Goal :

- Find the position of the blocks in the tray
- Result on the same rectangular pattern

Strategy:

1. Tray extraction and homography
2. Block extraction inside the tray
3. Display the result

Plan

1. Tray extraction & homography
2. Blocks extraction
3. Application
4. Results & recommendations

Tray extraction

Tray extraction algorithm consists of two main parts:

- finding coordinates of the tray 4 corners
- homography

We are going to talk about both of it consistently.

Tray localization algorithm overview

Main steps of the tray corners finding algorithm are:

1. Edge detection
2. Finding long enough line segments
3. Finding points of intersection

Let’s consider each step more precisely using the example image.

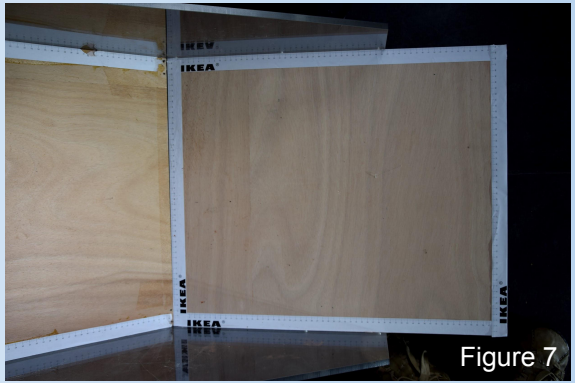


Figure 7

Edge detection using Canny algorithm

Canny edge detection[1] consists:

1. Smoothing (to suppress noise)
2. Finding gradients intensities and directions
3. Extraction candidates to be edges and its adjustment

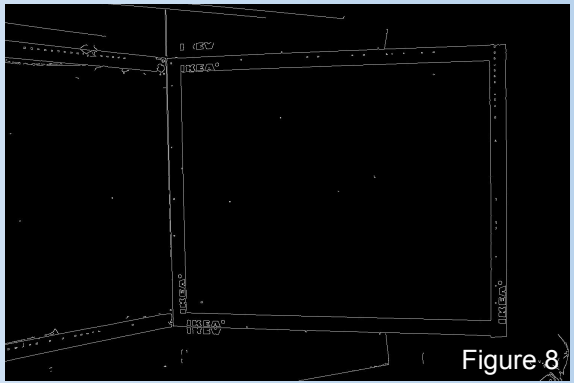


Figure 8

[1] Canny, J., *A Computational Approach To Edge Detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 8(6):679–698, 1986.

Finding long enough line segments

Special type of Hough transform[2] - probabilistic Hough transform[3] allows us to extract the most probable line segments longer then chosen minimal length.

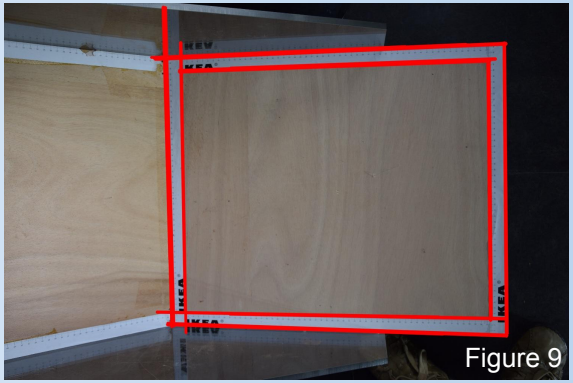


Figure 9

[2] Hough, P.V.C. Method and means for recognizing complex patterns. U.S. Patent 3,069 654, December 18, 1962

[3] N. Kiryati, Y. Eldar, A.M. Bruckstein, A probabilistic Hough transform, Pattern Recognition, 1991, https://doi.org/10.1016/0031-3203(91)90073-E.

Binarization and skeletonization

We have already extracted all necessary information from image and now we want to work only with obtained line segments, what is why we do such binarization: we drawn each pixel corresponding to found on previous step line segments with 1 on the image filled with 0.

Skeletonization is reducing binary objects to 1 pixel wide representations without losing connectivity.

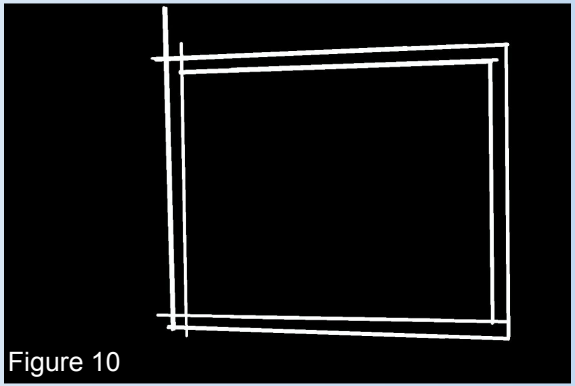


Figure 10

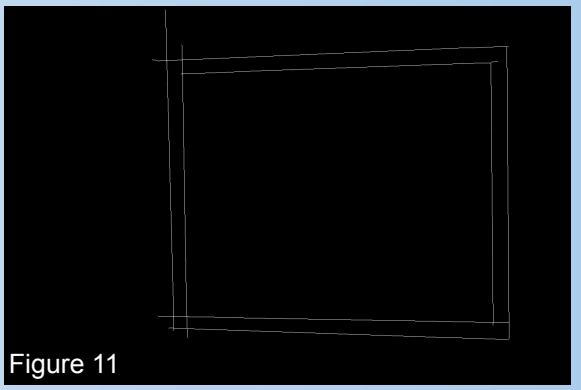


Figure 11

Corners finding

Remaining part of the algorithm consists only in:

1. Finding straight lines going through obtained thin lines related to inner and outer sides of white tape
2. Finding its points of intersection
3. Extracting from the obtained set of points correct points corresponding to the corners

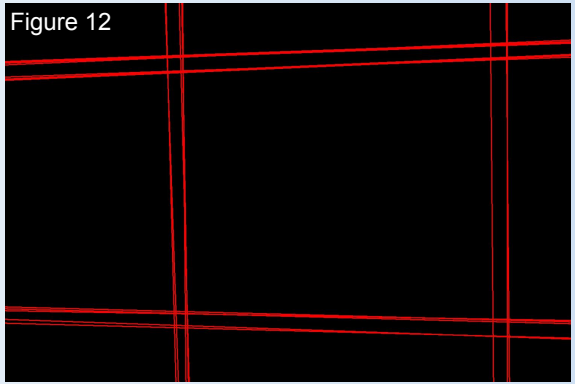


Figure 12


Figure 13

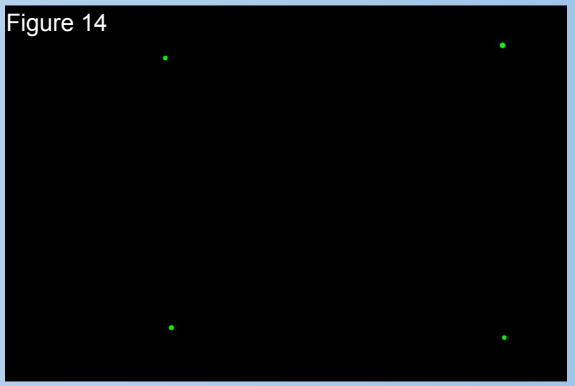


Figure 14

Good cases

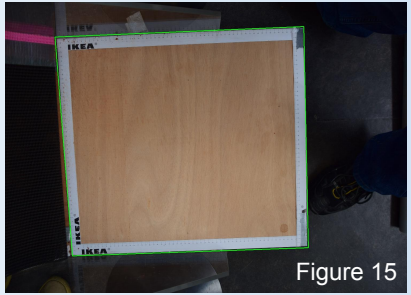


Figure 15

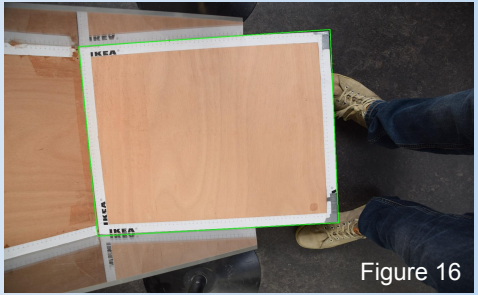


Figure 16

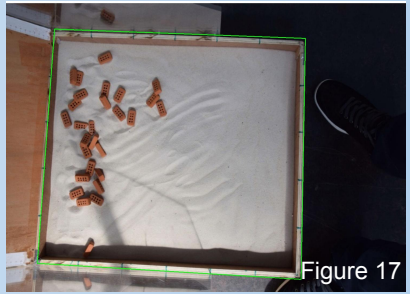


Figure 17

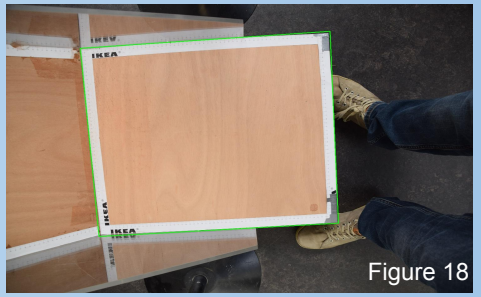


Figure 18

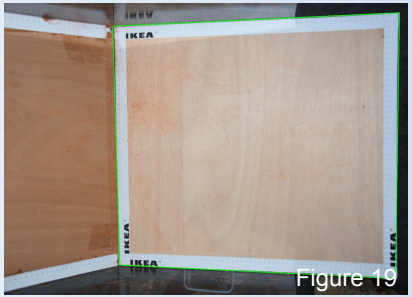


Figure 19

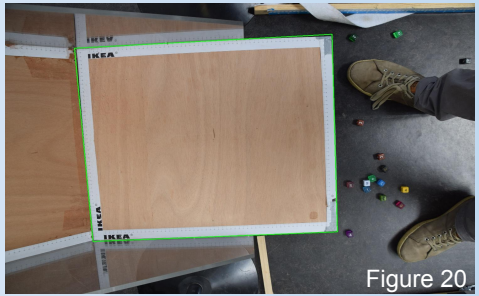


Figure 20

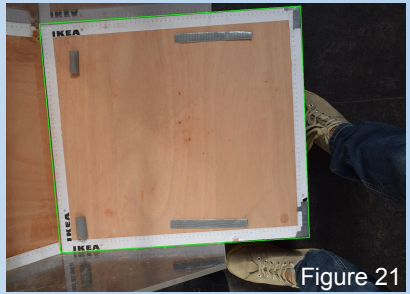


Figure 21


Figure 22

Bad cases

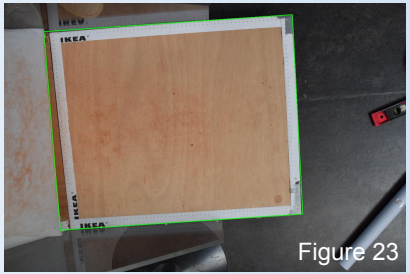


Figure 23

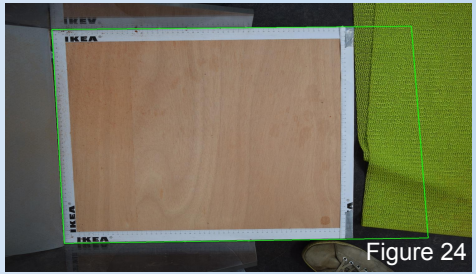


Figure 24

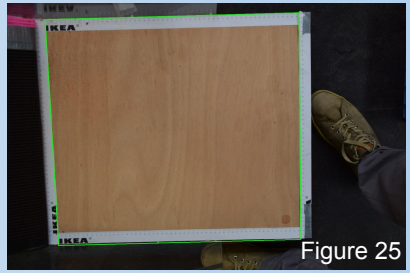
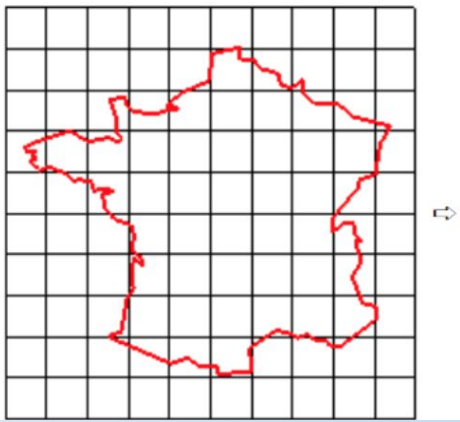


Figure 25


Figure 26

Homography



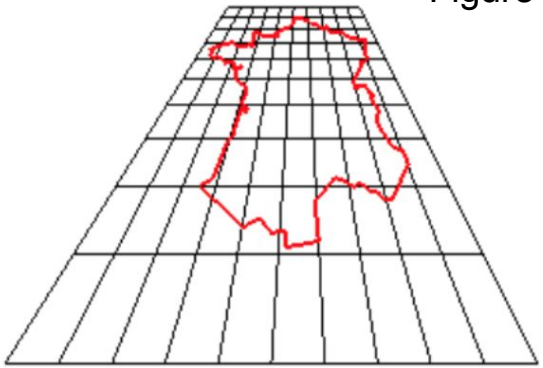


Figure 27

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

Converting *from* homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

Figure 28

Block extraction

- ➔ At this step, we consider that the tray is well extracted. If the previous step failed, we can select manually the tray.

- ➔ The more crucial step consists in determining a good mask for the blocks.

- ➔ Once the mask is created, we find the contours on the mask. If the mask is correct, each contour will correspond to a block.

First mask.

Many strategies for this mask but for this presentation we only consider the mask InRange based on HSV color space.

This mask is simply 3 low thresholds and 3 high thresholds for the 3 components of the HSV space.

The more efficient way consists in determining by hand these thresholds for a set of photos, because the conditions change for each sets.

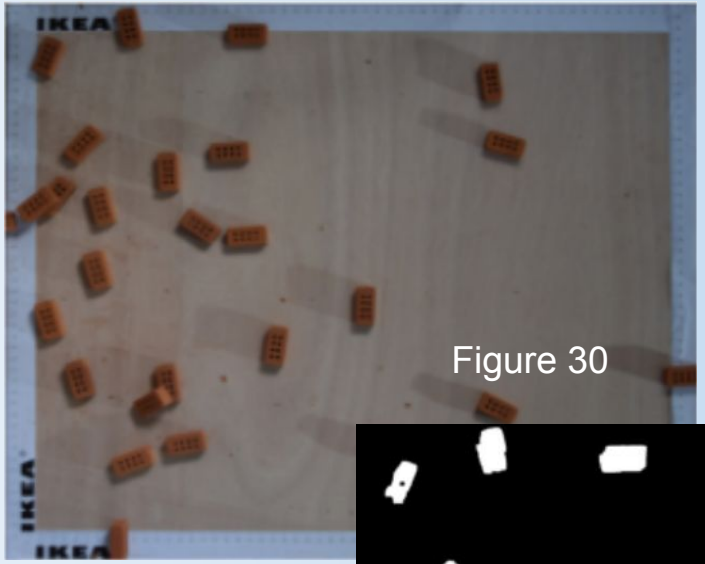


Figure 30

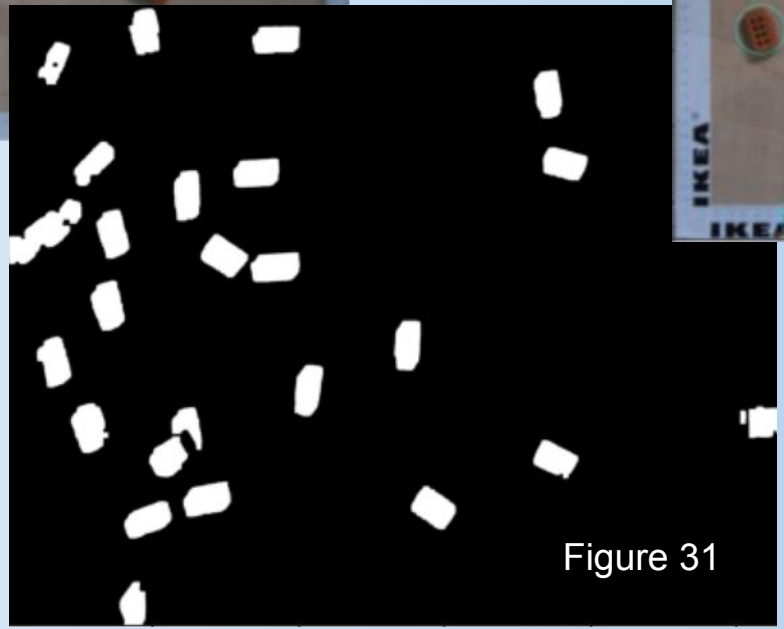


Figure 31

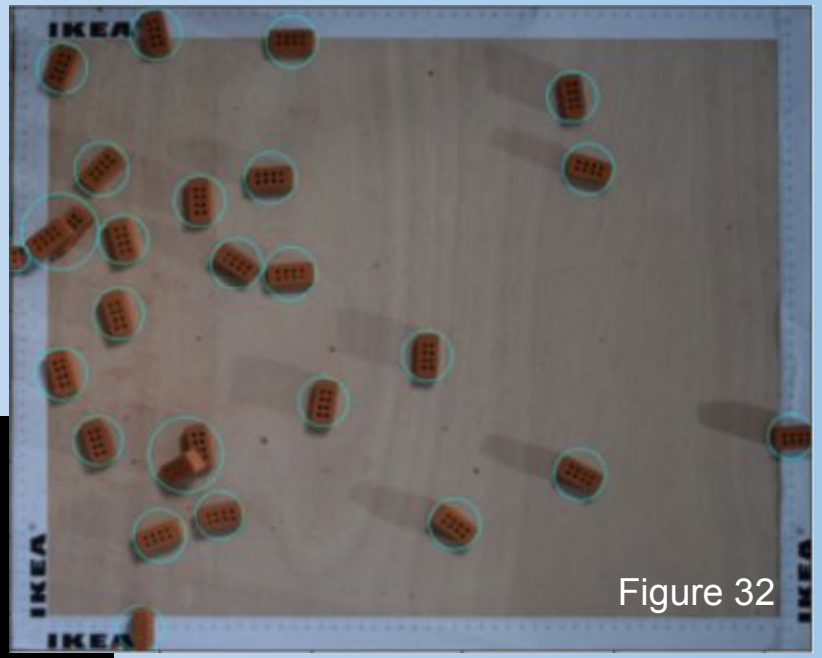


Figure 32

Second mask

- ➔ The first mask is not sufficient because a circle may correspond to many blocks. So a new mask is necessary.

- ➔ The first mask is applied to the original image, then Canny edges detector allows us to find border between each block. We add these border to the previous mask.

- ➔ To increase the quality of the new mask, we can detect the little dark points of the blocks (thanks to Hough circle algorithm) and remove them from the new mask.

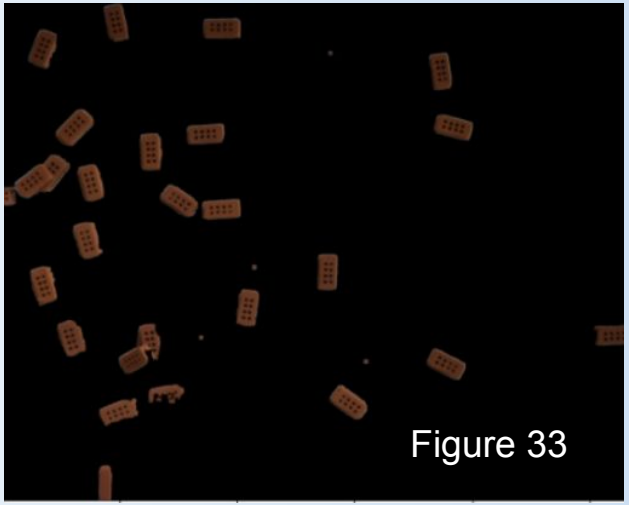


Figure 33

Canny

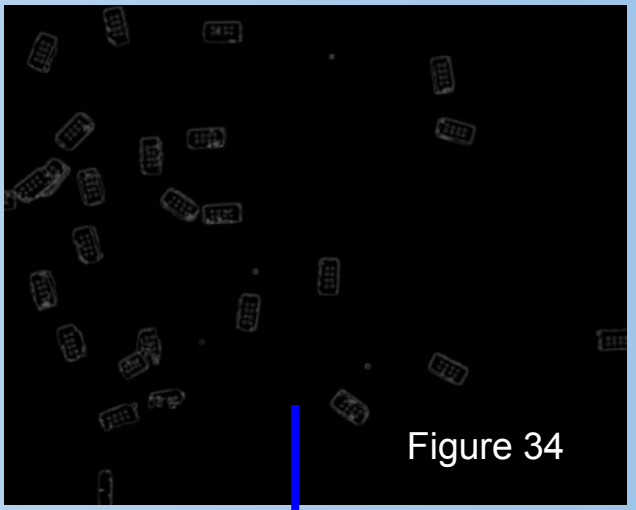


Figure 34

Hough circles

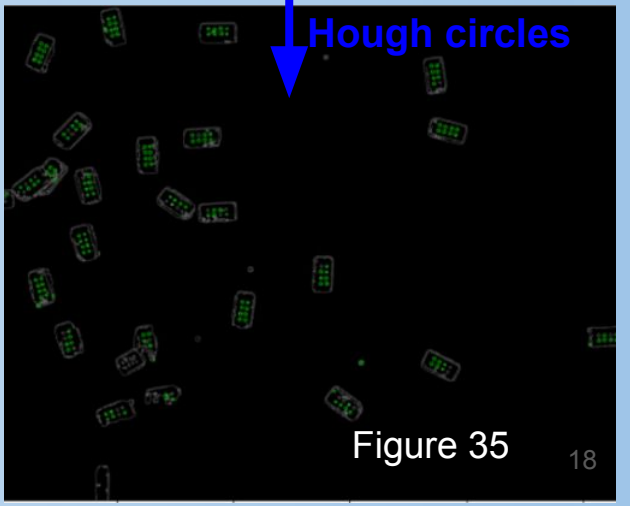


Figure 35

Circles deletion

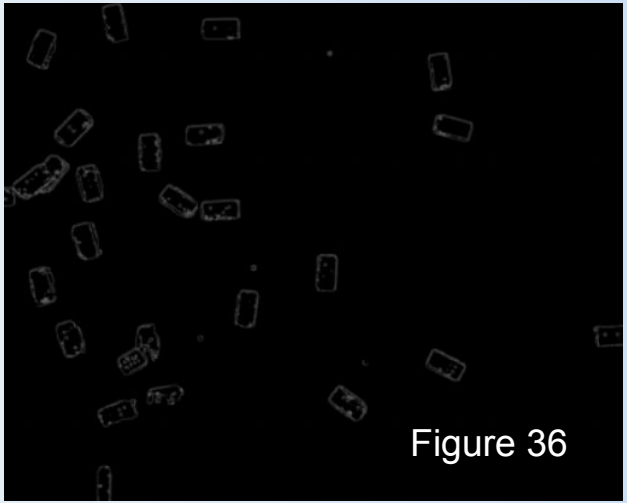


Figure 36

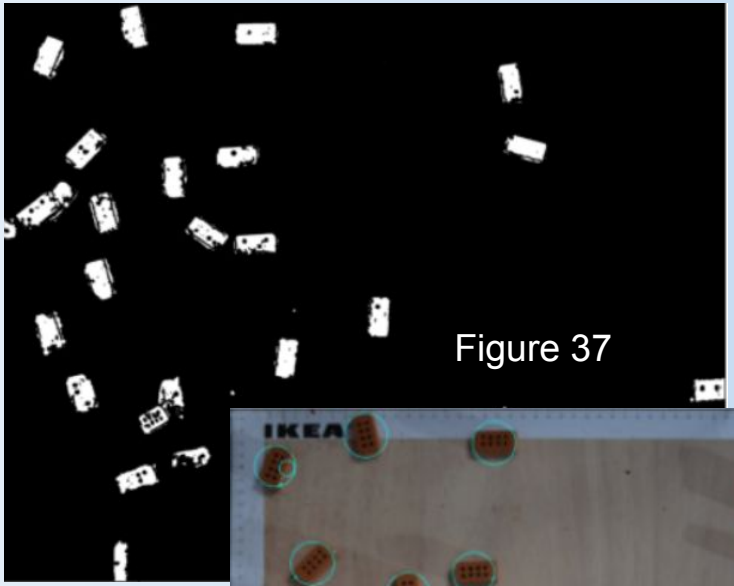


Figure 37

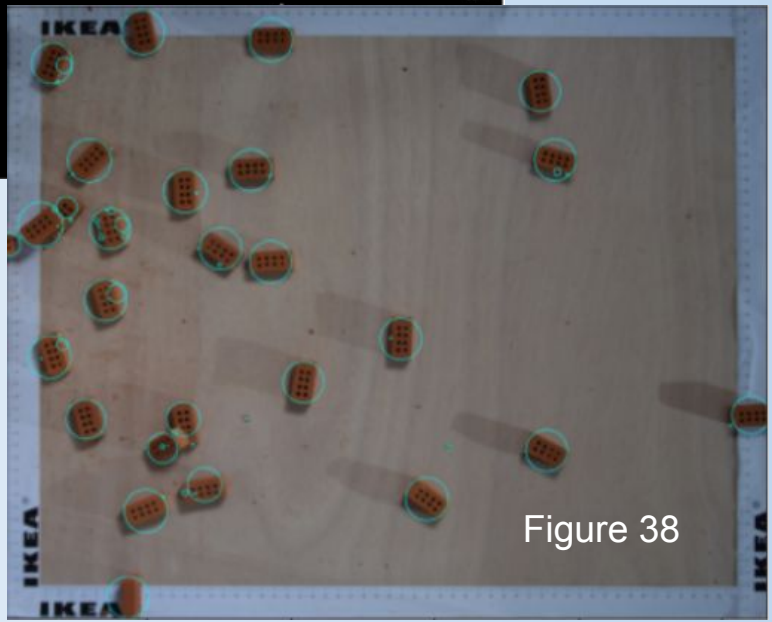


Figure 38

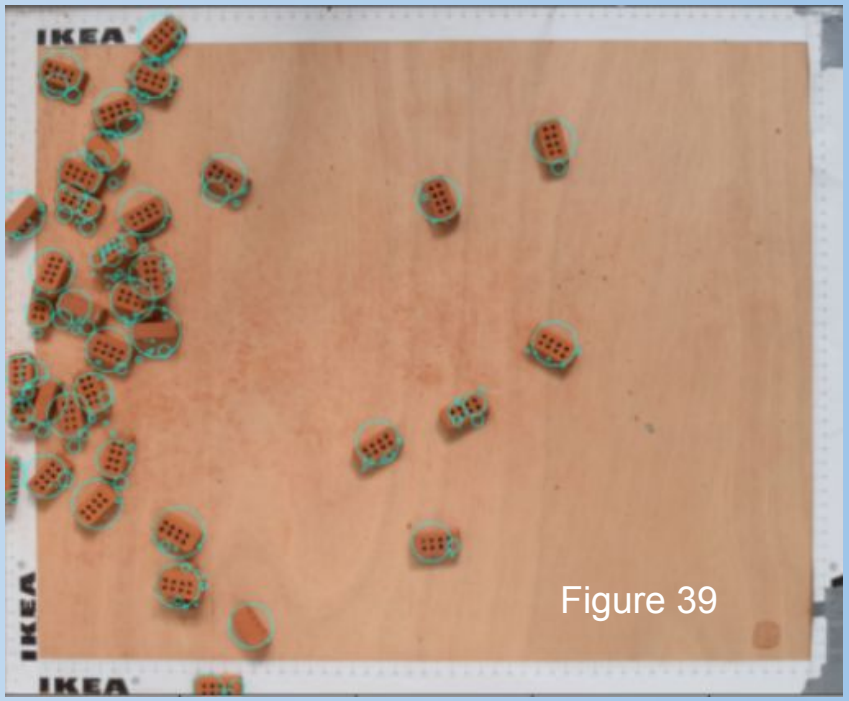


Figure 39

Circles filtering

- ➔ The new filter make appear too many contours and therefore too many circles. However it appears that a clever filtering of them can lead to good results.
- ➔ Filtering :
  - Deletion of very little circles.
  - Deletion of circles inside a bigger circle.
  - Reunion of very near medium circles into a bigger circle.
  - If a circle exceed a limit radius, we consider than it contains 2 blocks. Thus we delete it and we take the 2 bigger circles which were inside it.

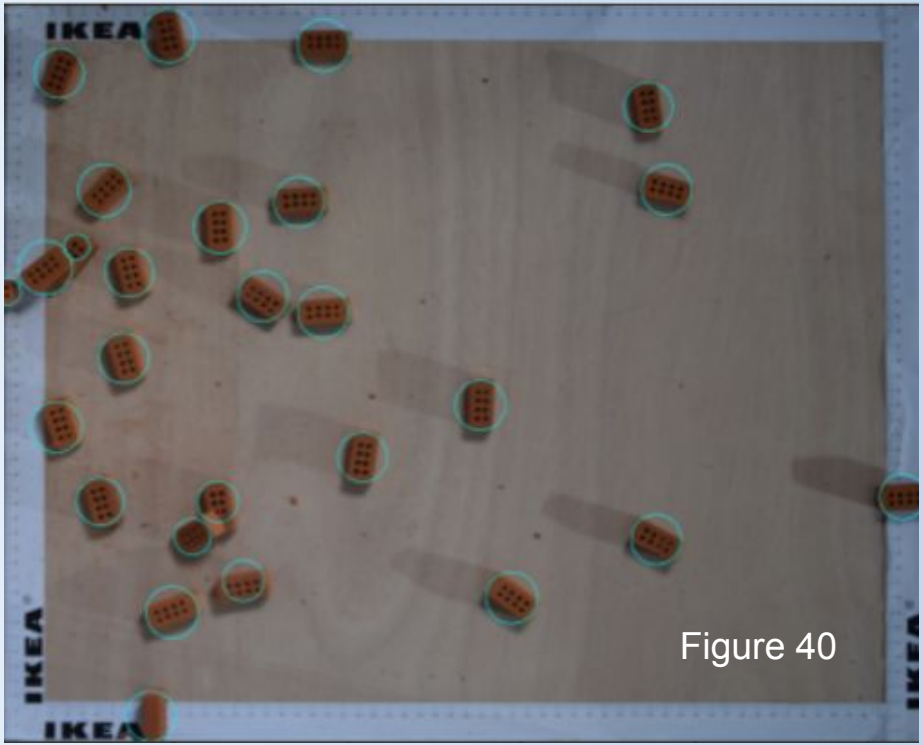


Figure 40

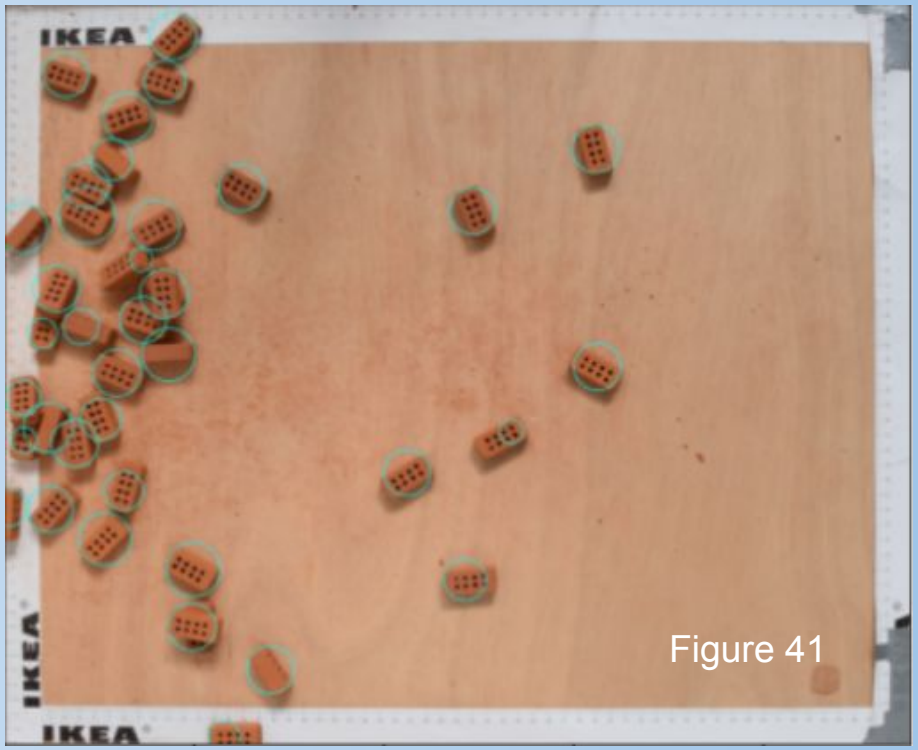
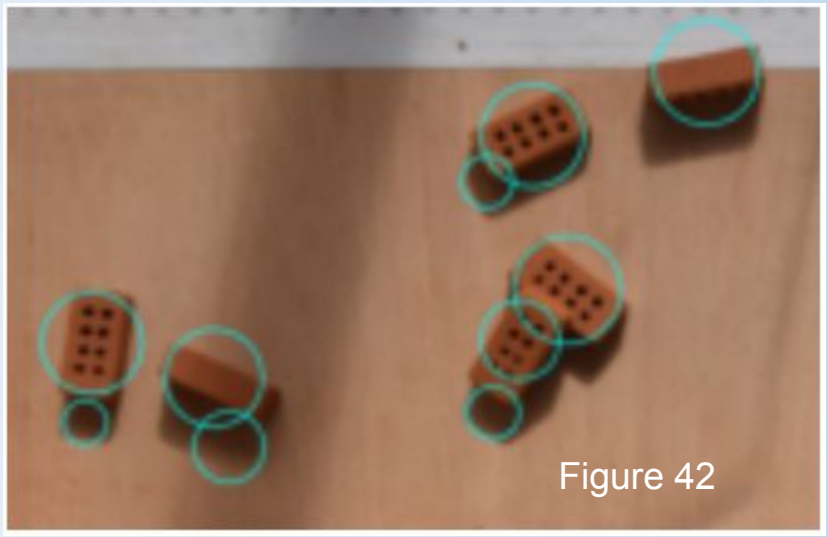
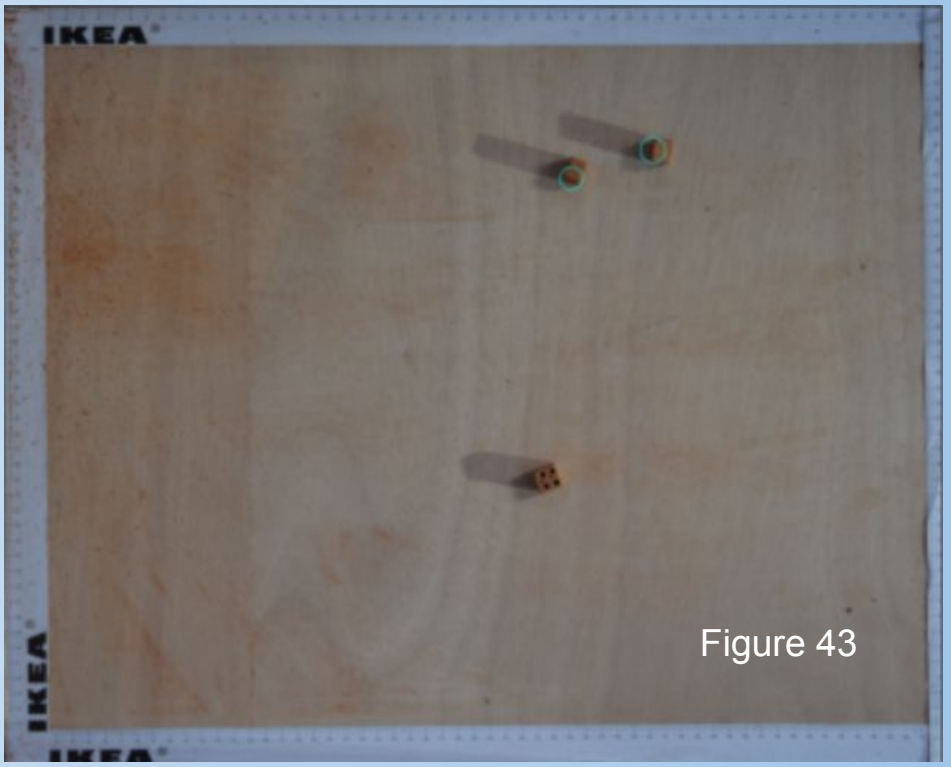


Figure 41

Limitations

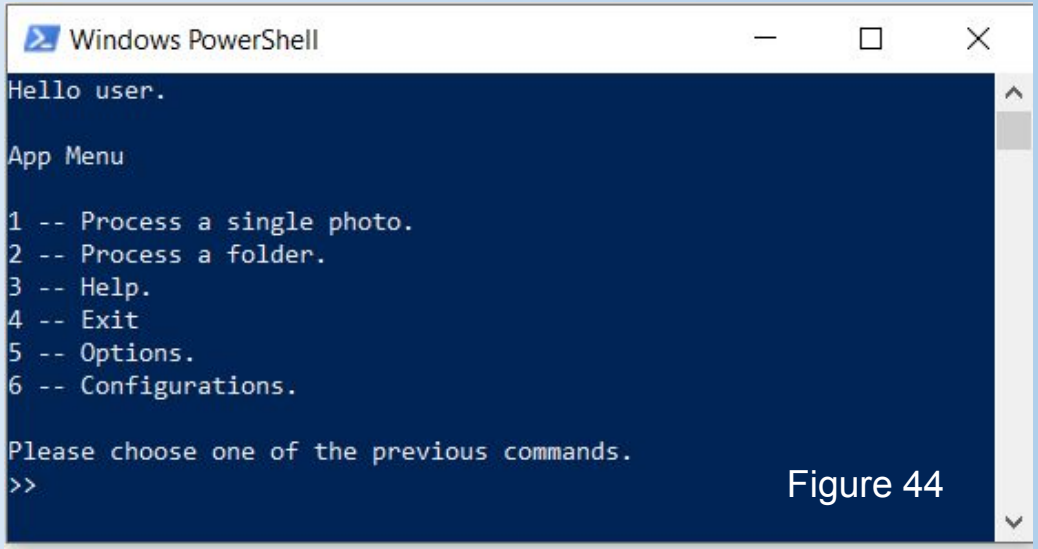


Shadows



Low brightness

Application



Menu of the app.

Options

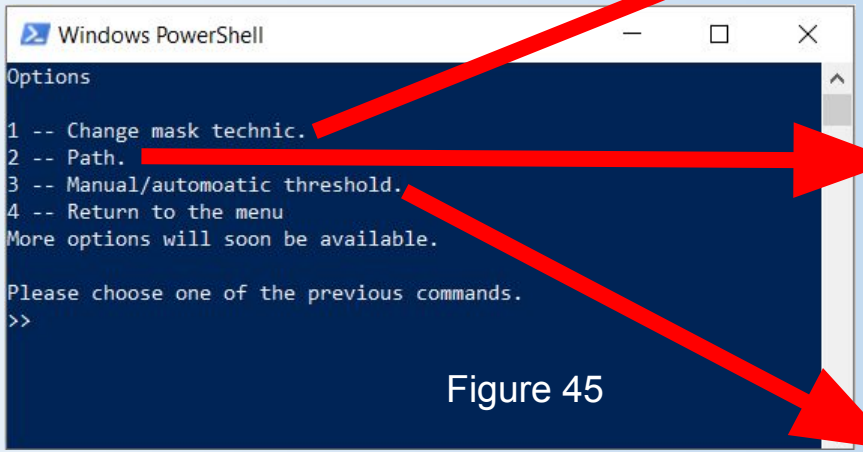
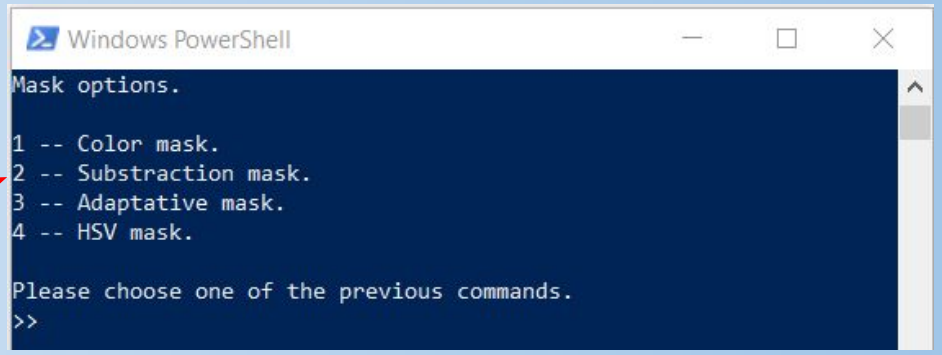
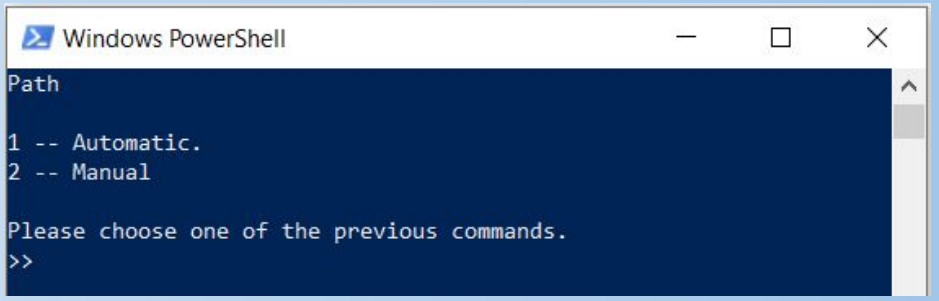
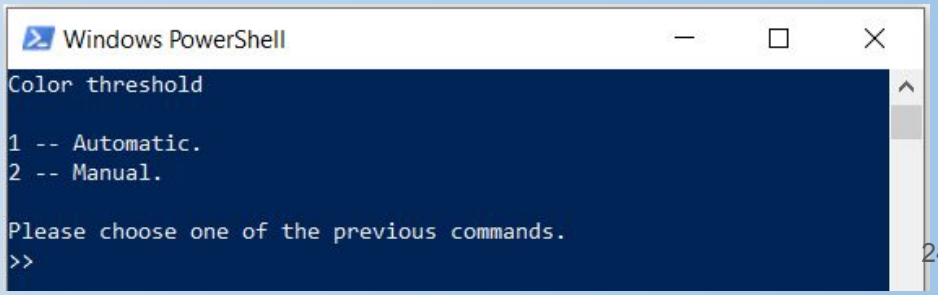


Figure 45







Configurations

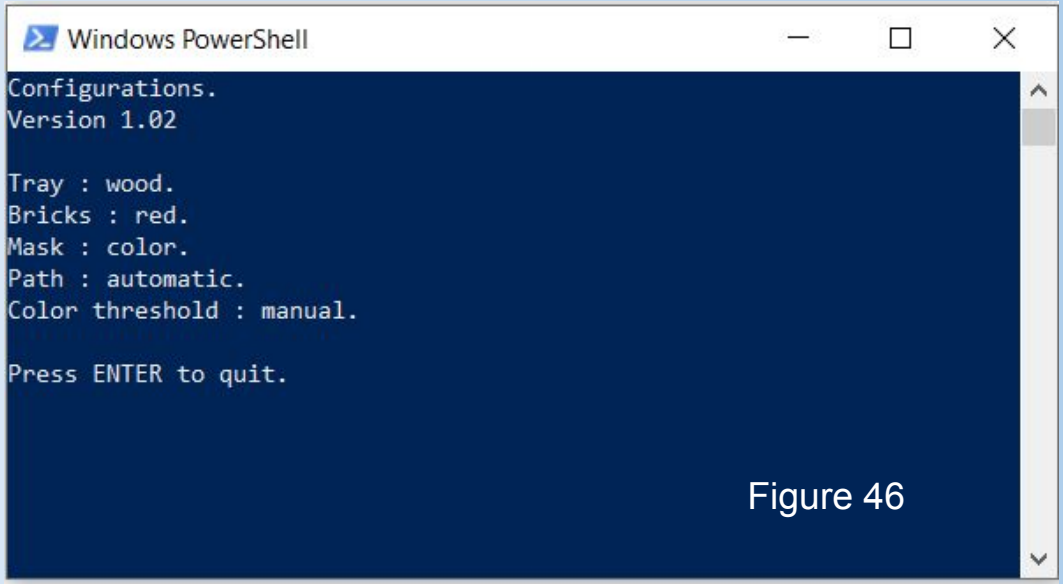
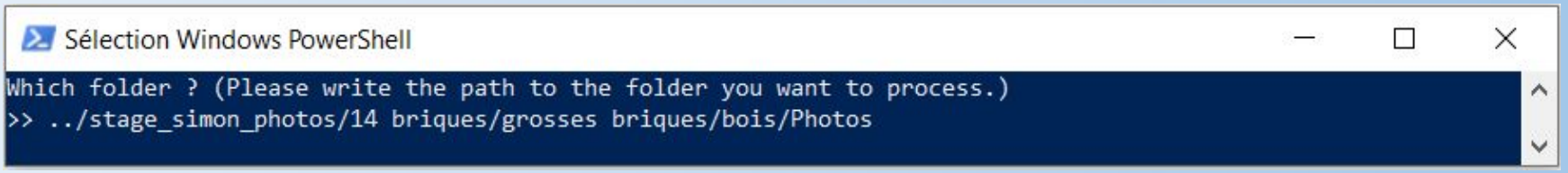


Figure 46

You can check the current options in configurations.

Process a folder



The app tries to detect the 4 angles and ask the user to confirm or infirm.

If the user refuse, he can click himself on the 4 angles.

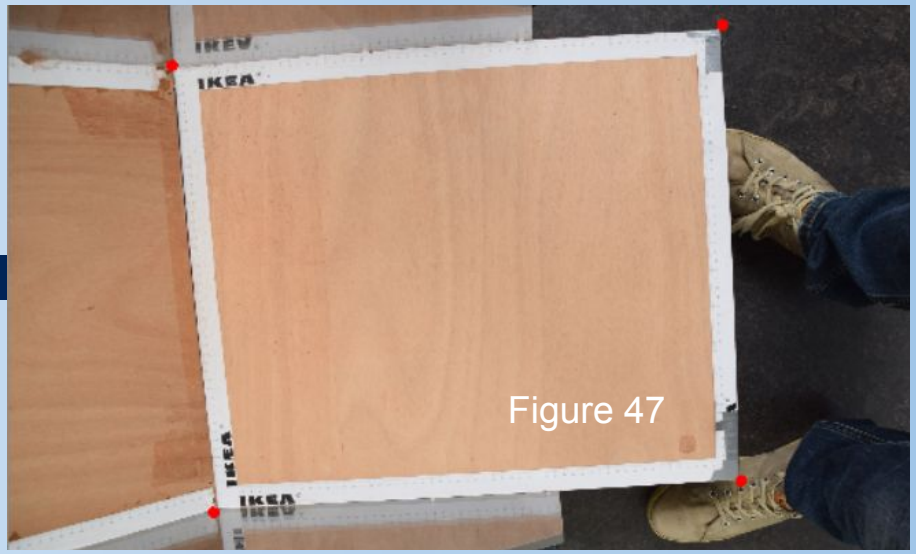


Figure 47

```
Corners successfully saved.
You have to choose manually the color to detect.
Please click at least 10 times on bricks.
```

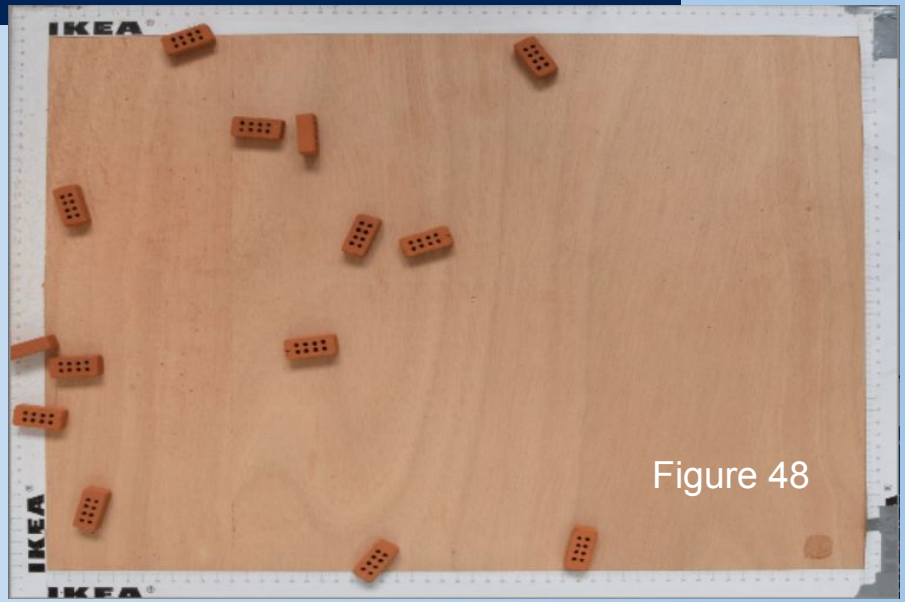


Figure 48

In the case of manual threshold, the user must click on the bricks so that the app know the color of the bricks.

Then the algorithm will both save photographies with detected blocks and save the associated csv file.

Some results

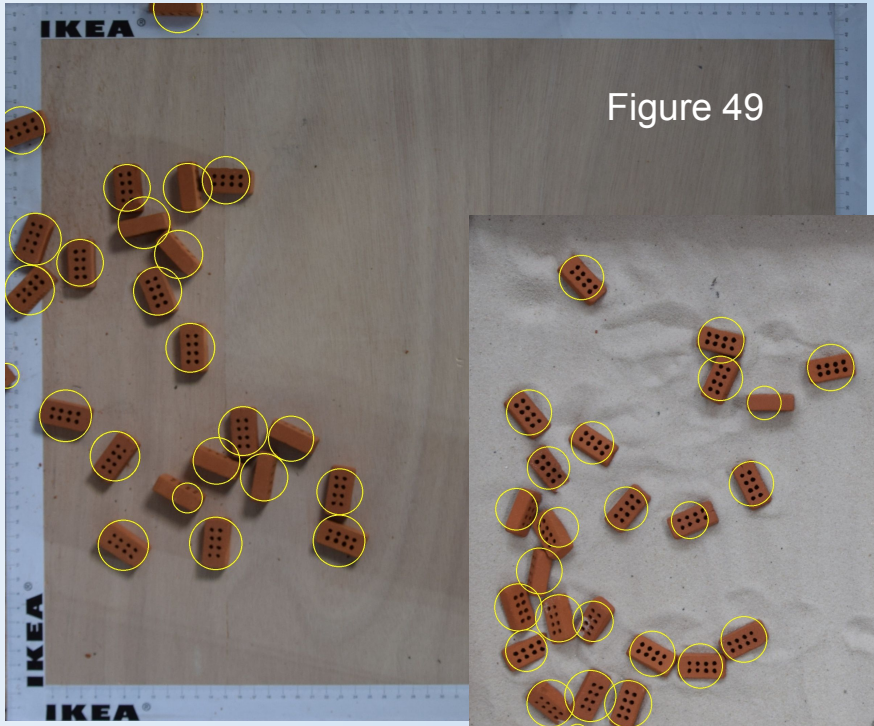


Figure 49

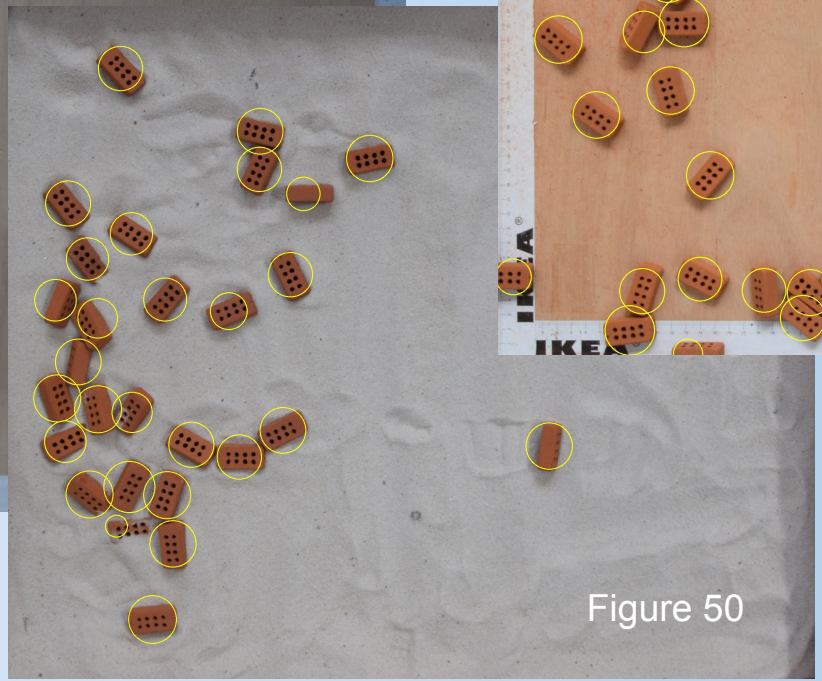


Figure 50

Figure 51

Results and Recommendation for the shooting

Our algorithm was evaluated on 50 pictures. On each pictures 14 big blocks were thrown on a wooden tray.

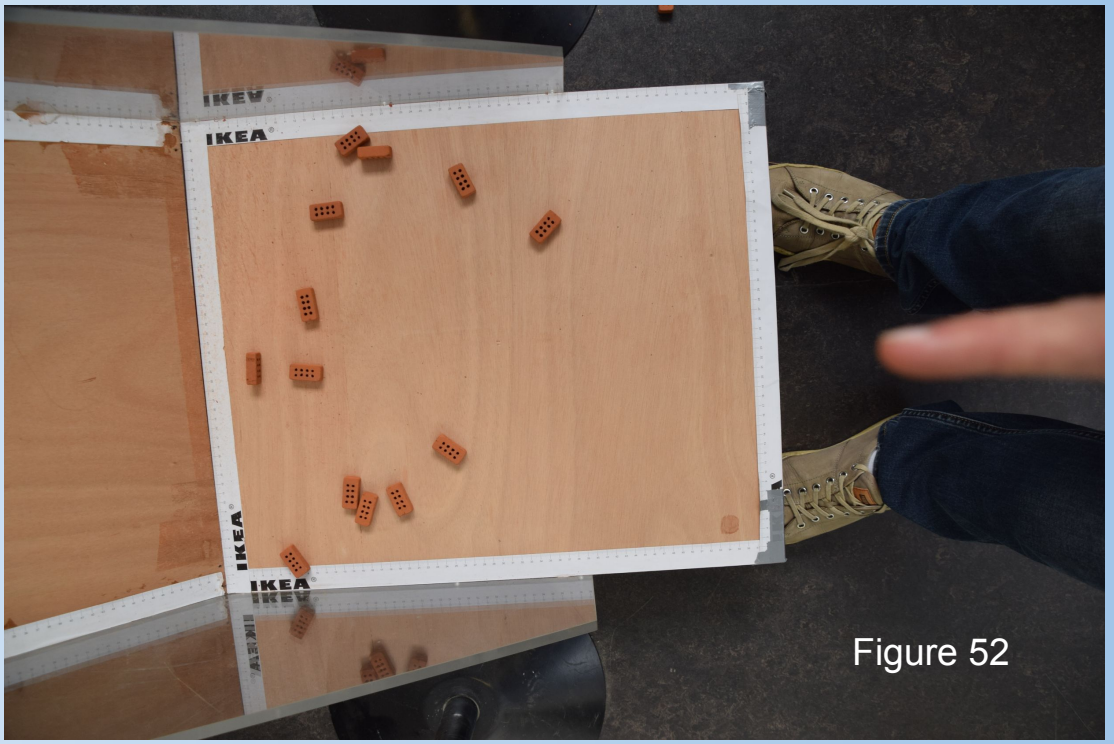


Figure 52

Data

	0	1	2	3	4	5	6	7	8	9	...	21	22	2
0	1	14	0	35.197523	16.519179	19.998295	11.941713	4.513883	23.599115	27.769780	...	17.336349	19.910980	20.53501
1	2	14	0	31.298408	27.890697	41.458567	25.225096	21.888791	22.081376	31.207467	...	31.524062	33.083644	43.24920
2	3	14	1	36.467089	28.325117	19.641443	17.431282	14.230665	11.643330	10.852407	...	10.345129	20.140925	27.18834

Figure 31

Columns :

- 0 represents an index
- 1 represents the number of block thrown
- 2 represents the number of block outside the tray
- 3 - 16 X-coordinates
- 17-30 Y-coordinates

Number of block outside the tray

- 66% of accuracy
- 33 pictures

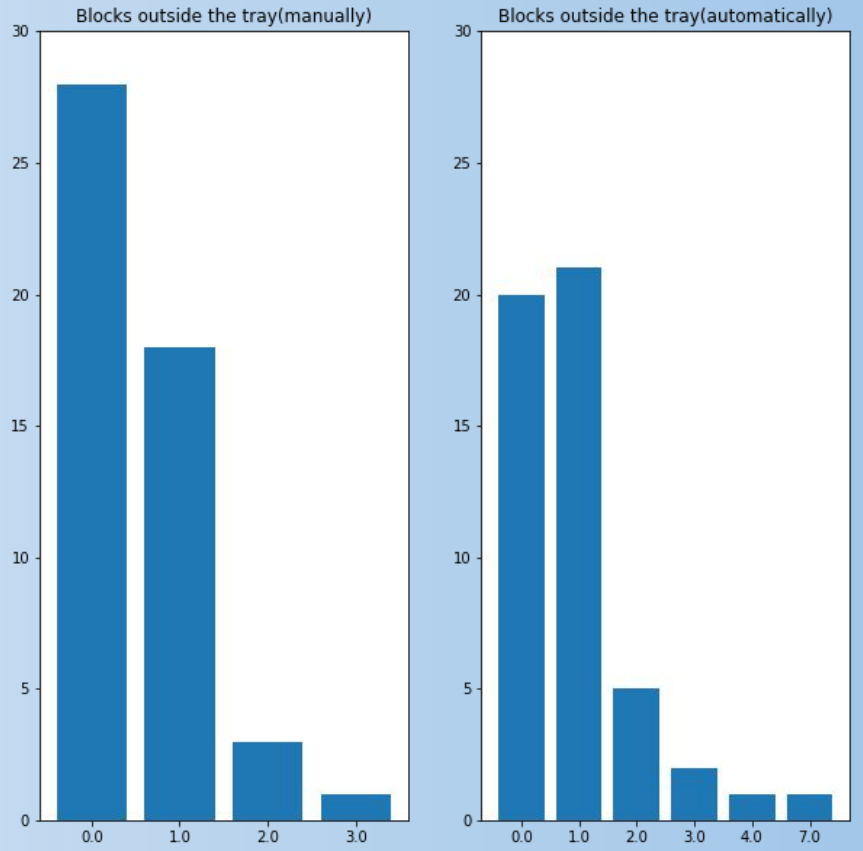


Figure 54

Influence on the distance of the automatic measurement

Data : Concatenation of the concatenation of X-coordinates for each image without the 0.

The two dataset follows the same distribution.

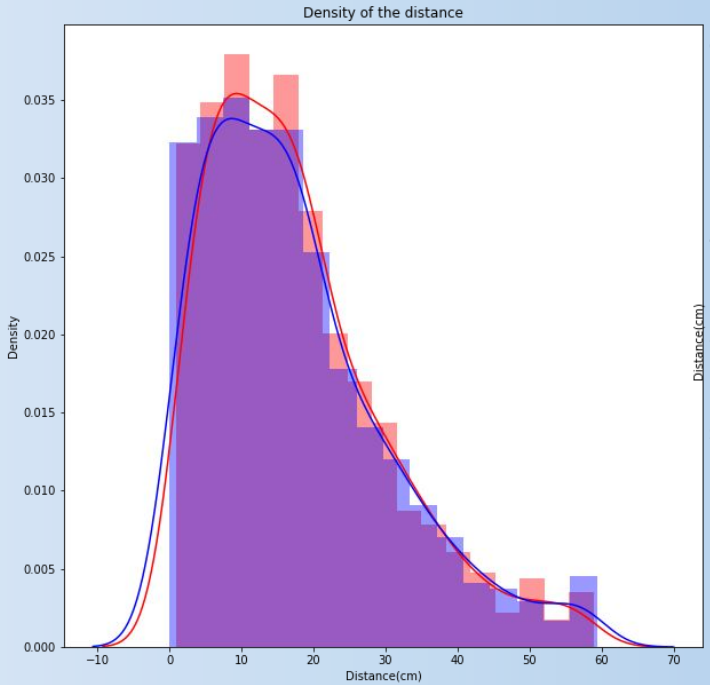


Figure 55

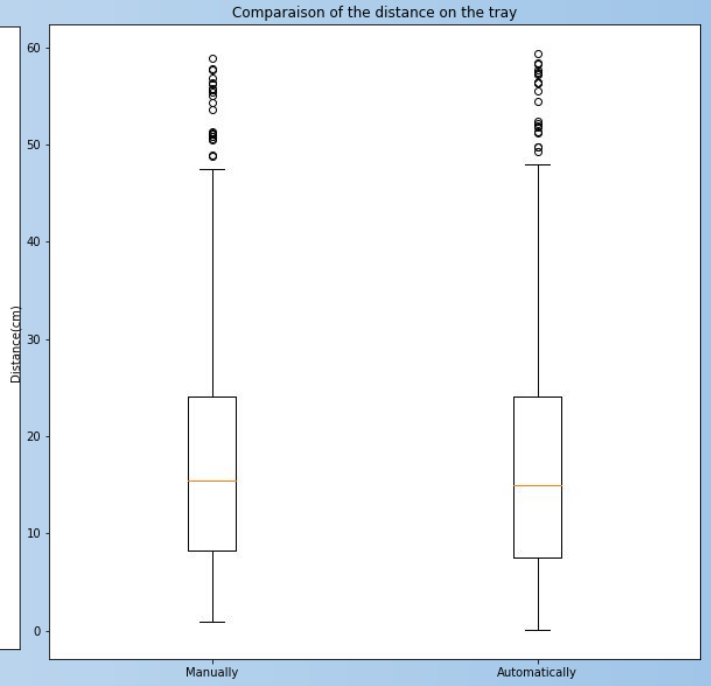


Figure 56

Analysis of the error

During the error analysis we will only look at the images where the right number of bricks have been found.

Data:

- Mean : 0.33
- Variance : 0.58

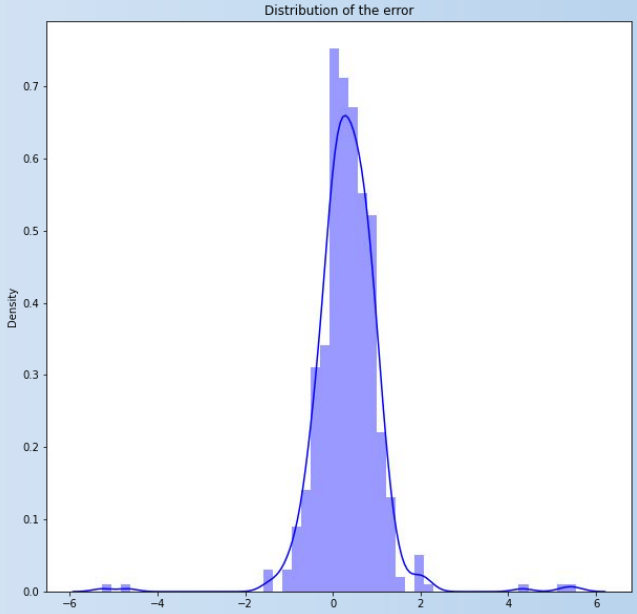


Figure 57

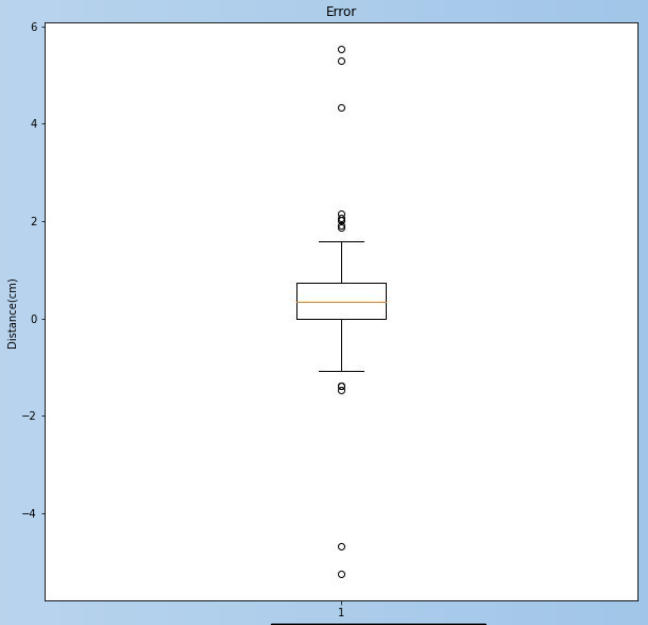


Figure 58

Analysis of the error for each picture

Mean absolute error = 0.73 cm

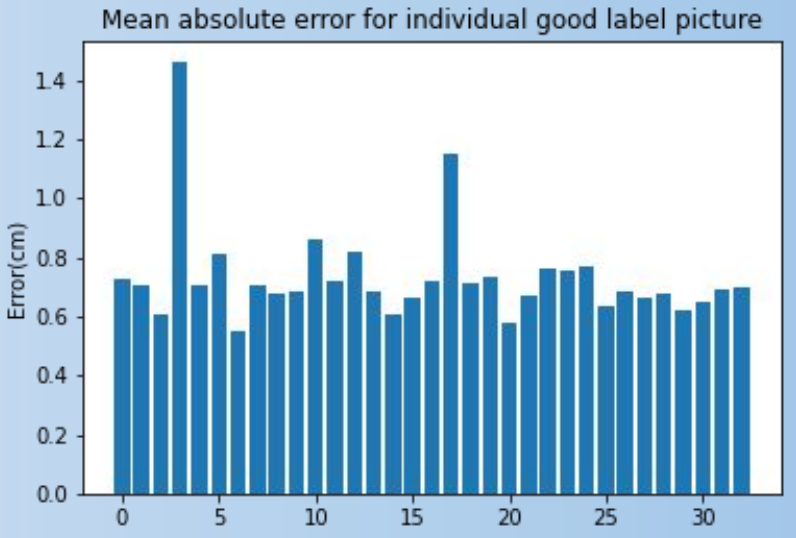


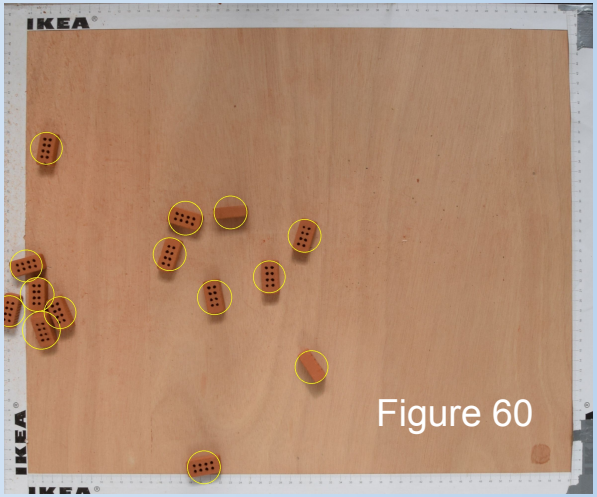
Figure 59

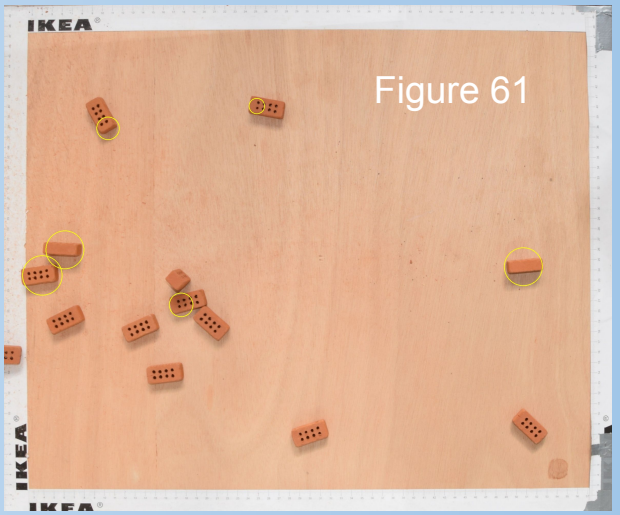
Recommendation for shooting

We have given a lot of comments and advice in the report on how to take the pictures to improve the results.In this last part, we will only give in relation to the shooting done on the previous dataset.

Recommendation : Environment

The brightness should be the same for all images in the same folder.





No objects other than bricks must appear on the tray.

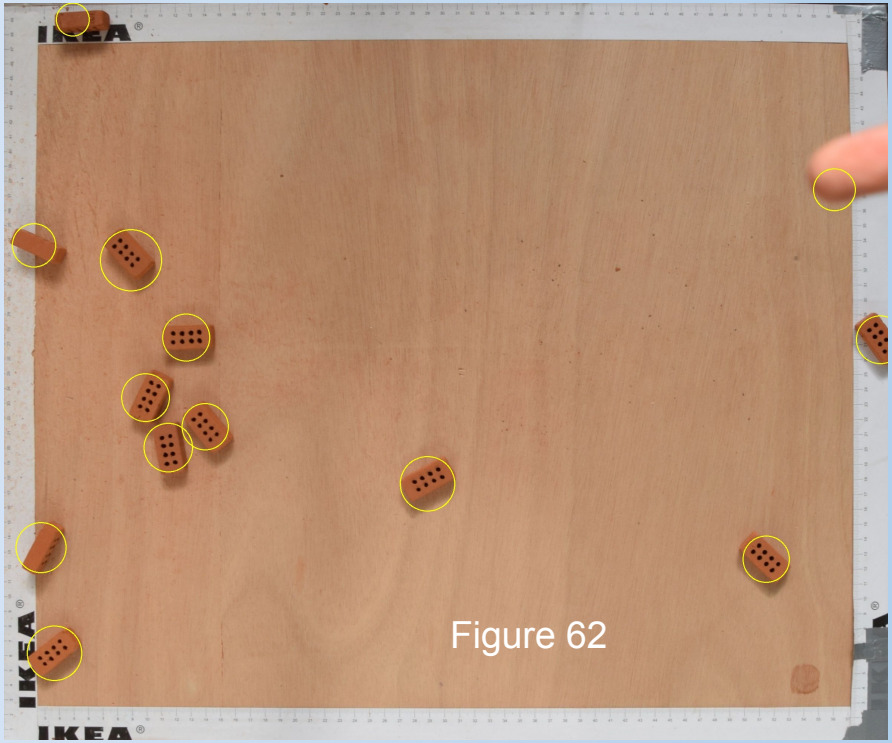


Figure 62

Thank you for your time and your attention