Make It Aesthetic

Anna-Maria Hohmann
Konrad Kring
Meret Lindanis

2nd presentation
05.11.2019

About Us

Meret

- computer science
- medical technologies

Anna

- business informatics
- project management

Konrad

- pedagogics
- music

Goals of Our Project

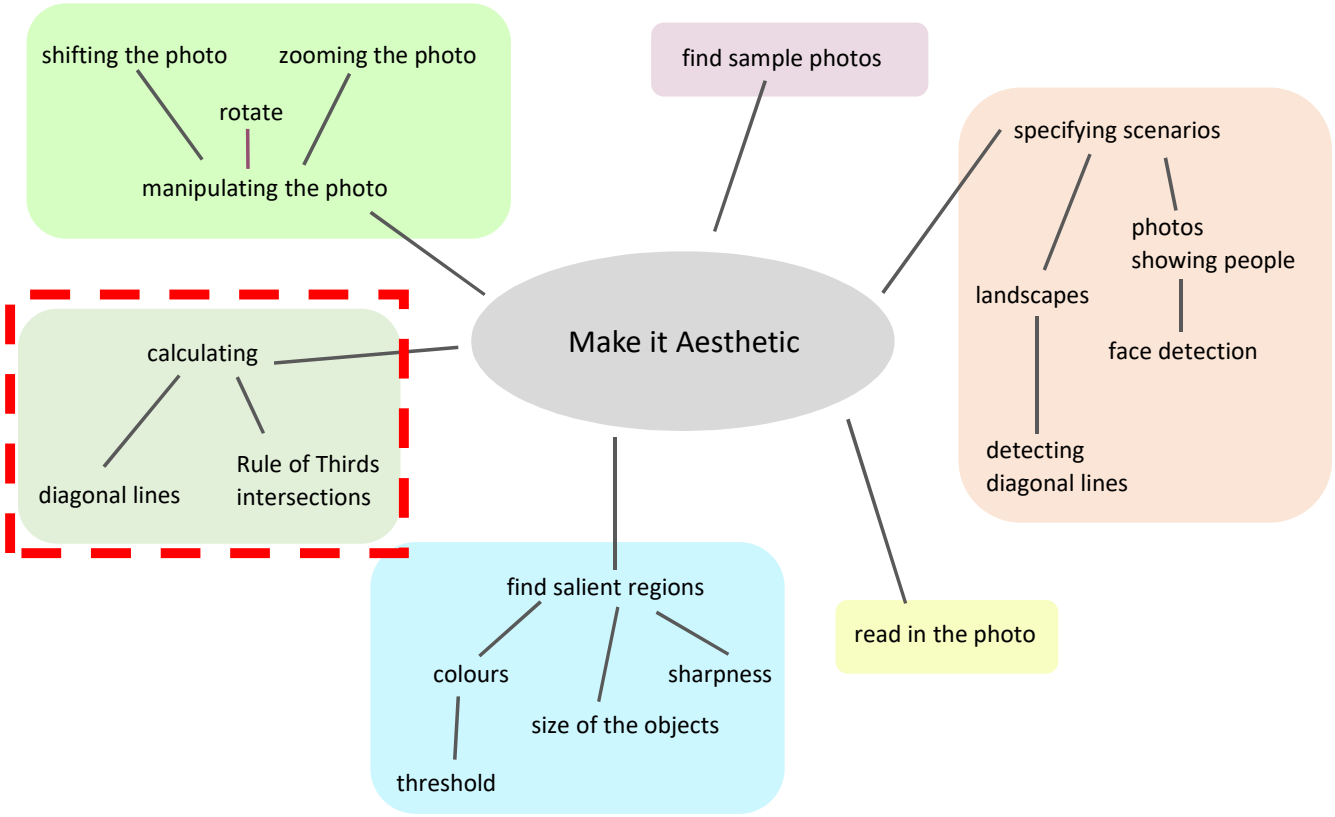
our motivation:

- interested in photography
- opening aesthetic photography to the public
- simplifying the aesthetic photography for the user
- being able to save every moment in beautiful photos
- bringing this knowledge into school

Goals of Our Project

- make given photos aesthetic
- by zooming, rotating or cropping the photo
- selecting the guideline the photo should follow

Milestones


shifting the photo
zooming the photo
rotate
manipulating the photo
find sample photos
specifying scenarios
photos
showing people
landscapes
face detection
detecting
diagonal lines
Make it Aesthetic
calculating
diagonal lines
Rule of Thirds
intersections
find salient regions
colours
sharpness
size of the objects
threshold
read in the photo

Use-Cases

- make photos more aesthetic for a photo album / website
  - wedding /big events --> people/creature
  - vacation --> landscape, buildings, people
  - art --> all categories
- let the algorithms editing all your photos instead of doing it manually

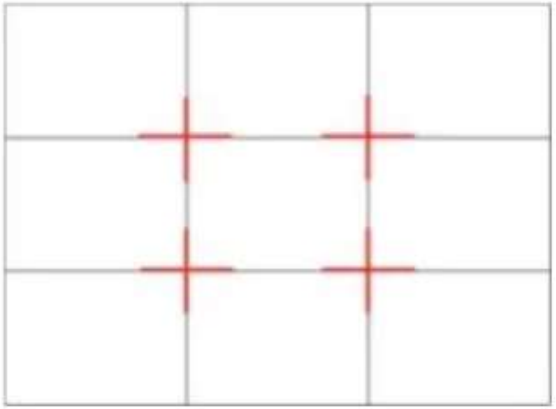
Scenarios

- using clear colour contrast first
  - bright background, dark main object
  - dark background, bright main object
  - only one object
- adding more and more complicated scenarios
  - more objects
  - fewer colour contrasts
  - i.e. bright background, bright main object
  - and vice versa

Focus

Rule of Third (RT):

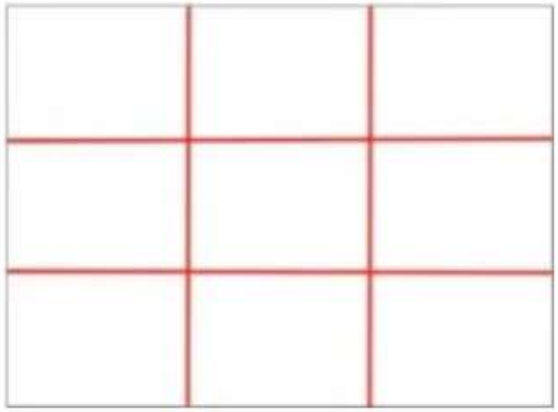
- breaking image down into thirds
- horizontally / vertically = 9 parts
- gives four important parts for placing points of interest in photo



Focus

Rule of Third:

- gives also four important lines
- useful positions for elements in photo



Implementation

Importing image to our notebook

```
! wget -q https://raw.githubusercontent.com/.../snail.JPG
img_snail = cv2.imread("snail.JPG")
img_snail = cv2.cvtColor(img_snail, cv2.COLOR_BGR2RGB)
plt.imshow(img_snail)
```

```
<matplotlib.image.AxesImage at 0x7fc287d01c18>
```

Implementation

Function for drawing the RT-lines

```python
def draw_rule_of_thirds(img, line_color, line_width):
  size = img.shape
  height = img.shape[0]
  width = img.shape[1]

  #calulating the line for the rule of thirds
  third_of_height_1 = height // 3
  third_of_height_2 = (height // 3) * 2
  third_of_width_1 = width // 3
  third_of_width_2 = (width // 3) * 2

  #drawing the line on the photo calculated above
  #drawing the horizontal lines
  img = cv2.line(img, (0, third_of_height_1), (width, third_of_height_1), line_color, line_width)
  img = cv2.line(img, (0, third_of_height_2), (width, third_of_height_2), line_color, line_width)
  #drawing the vertical lines
  img = cv2.line(img, (third_of_width_1, 0), (third_of_width_1, height), line_color, line_width)
  img = cv2.line(img, (third_of_width_2, 0), (third_of_width_2, height), line_color, line_width)
  return img
```

Implementation

Calculating the height and width of the image

```
height = img.shape[0]
width = img.shape[1]
```

Implementation

Calculating the position of the lines

```
#calulating the lines for the rule of thirds
third_of_height_1 = height // 3
third_of_height_2 = (height // 3) * 2
third_of_width_1 = width // 3
third_of_width_2 = (width // 3) * 2
```

Implementation

Drawing all lines

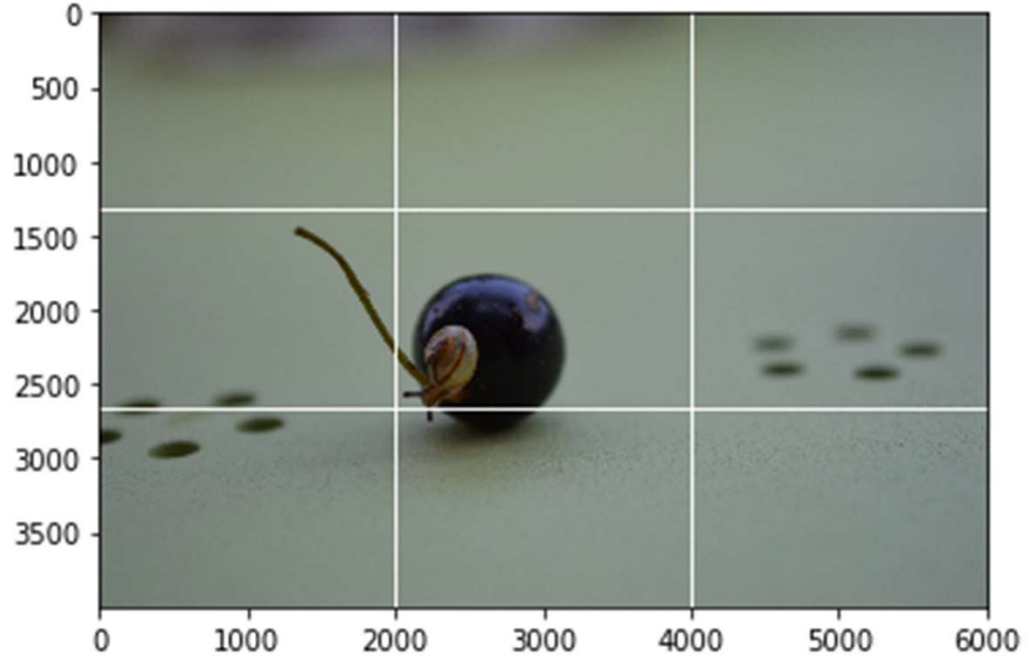
```
#drawing the lines on the photo calculated above
#drawing the horizontal lines
img = cv2.line(img, (0, third_of_height_1), (width, third_of_height_1), line_color, line_width)
img = cv2.line(img, (0, third_of_height_2), (width, third_of_height_2), line_color, line_width)
#drawing the vertical lines
img = cv2.line(img, (third_of_width_1, 0), (third_of_width_1, height), line_color, line_width)
img = cv2.line(img, (third_of_width_2, 0), (third_of_width_2, height), line_color, line_width)
return img
```

Implementation

The returned image

```
draw_rule_of_thirds(img_snail, (255, 255, 255), 5)
plt.imshow(img_snail)
```

```
<matplotlib.image.AxesImage at 0x7f4af58a4ba8>
```

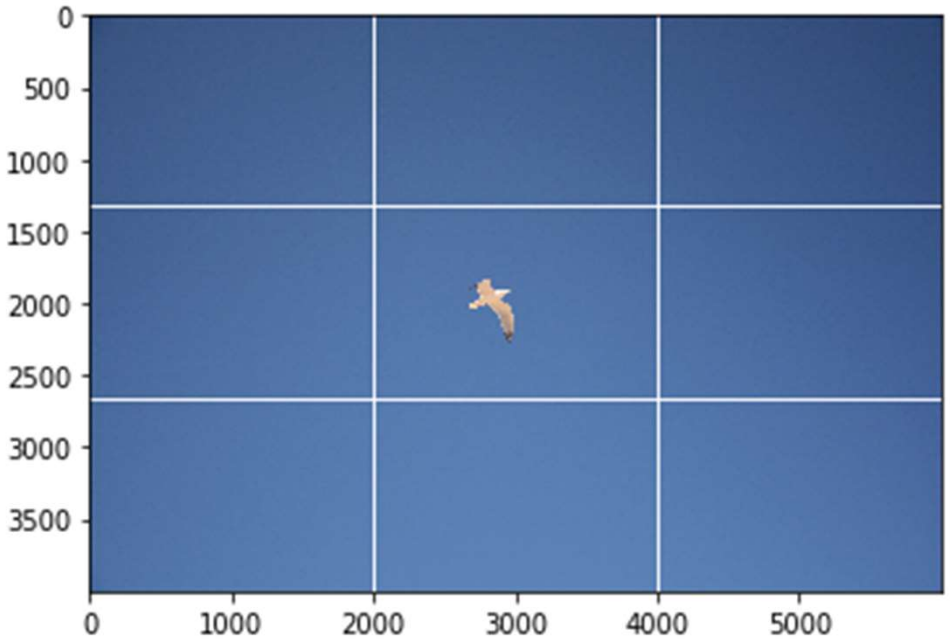


Implementation

Sample 1

```
draw_rule_of_thirds(img_seagull, (255,255,255), 17)
plt.imshow(img_seagull)
```

```
<matplotlib.image.AxesImage at 0x7fc285417eb8>
```

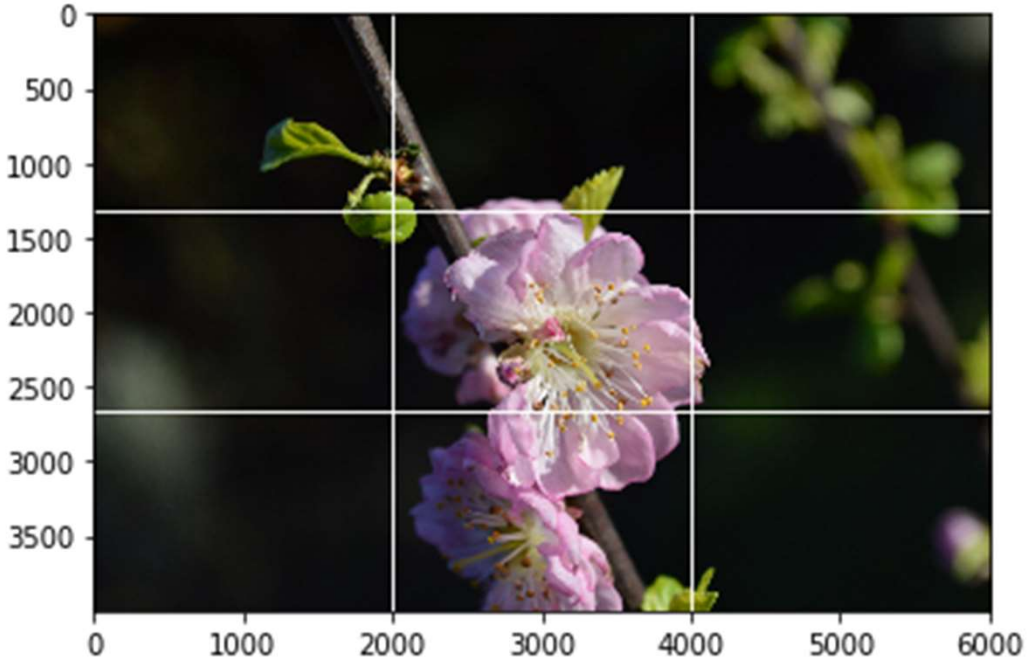


Implementation

Sample 2

```
draw_rule_of_thirds(img_flower, (255, 255, 255), 5)
plt.imshow(img_flower)
```

```
<matplotlib.image.AxesImage at 0x7f4af3f42390>
```



Challenges

- the RT-lines only appear with the right thickness value
- it is difficult to find the right thickness for the lines

Next Steps

- detecting the objects of interest in the photo
- calculating the points of interest respectively the intersections of the RT-lines

Sources

Literature / Internet

[1] https://digital-photography-school.com/rule-of-thirds/