Deep Learning for Computer Vision
Lab 02

Bolutife Atoki
bolutife-oluwabunmi.atoki@etu.u-bordeaux.fr
Université de Bordeaux

Abstract

This paper contains my solution and results of the second lab session, resources used are listed at the end of the paper

1 Introduction

The aim of this session was to compute the Gaze Fixation Density Maps / Wooding maps from gaze fixation points obtained on the MexCulture142 dataset. The obtained maps should then be overlayed on the images, using an appropriate colour map to highlight the fixation points on the images. The images in the MexCulture142 dataset were resized (preserving the aspect ratio), and displayed on a screen having a vertical resolution of 1200, and a height of 325mm.

2 Methodology

A Gaze Fixation Density Map (GFDM) / Wooding map is computed by initially obtaining gaze fixation data, which consists of points (x, y) in the image where an observer's gaze was fixated while viewing the image. These fixations could be computed from multiple observers and in the case of the dataset used, the images were resized and projected on a screen having a vertical resolution of 1200, and a height of 325mm, on which the gaze points were marked. Since the point of the GDFM is to represent the density of gaze fixations at different spatial locations in the image, convolution of the image with a Gaussian function is used to performed to represent these fixations in the same form and foveated angle (focus on object with smoothening / blurring of surrounding objects) as the eyes would see and interpret. To this end, a two-dimensional (isotropic) Gaussian function is used and centered on the gaze point, with an equal spread in both x and y dimensions controlled by a sigma value.

The GFDM's are computed for every fixation point in the image to obtain partial saliency maps, which are all added together to form a matrix (global saliency map), which represents the combined saliency information from all fixation points in the image. Finally, the global saliency map is normalized (by dividing it by its maximum value) to ensure all values range between 0 and 1.

2.1 GFDM Calculation

The Gaze Fixation Density Map (GFDM) is computed using:

$$S_g(X) = \frac{1}{N_{\text{obs}} \cdot M_{\text{fix}}} \sum_{i=1}^{N_{\text{obs}}} \sum_{m=1}^{M_{\text{fix}}} \delta(X - x_f(m)) * G_\sigma(X)$$

where:

X is a vector representing the spatial coordinates

$x_f(m)$ is the spatial coordinate of the mth visual fixation

M_{fix} is the number of visual fixation for the ith observer

N_{obs} is the number of observers

$\delta(.)$ is the Kronecker symbol, $\delta(t) = 1$ if t = 1, otherwise $\delta(t) = 0$

such that for each gaze record, a partial saliency map is computed by applying a two- dimensional Gaussian centered on the fixation. Then these partial saliency maps are summed to get a global saliency map. Finally, this global map is normalized by its maximum value.

2.2 Gaussian Spread Calculation

The spread of the Gaussian function around each fixation point is determined based on the properties of the screen and human vision. Wooding proposed to fix the Gaussian propagation at an angle of 2° (based on an imitation of the functioning of the fovea of the human eye which covers an area of 1.5° to 2° of the diameter in the center of the retina) It is calculated using the formula:

$$\sigma = R \cdot D \cdot \tan(\alpha)$$

where

R: Vertical resolution of the screen in pixels per mm.

D: A parameter that is typically set to about three times the height of the screen.

α: An angle (in radians) that represents the foveal angle

2.3 Partial Saliency Maps

The computation of the fixation density map (GFDM) provides information about the overall density of fixations across the image. This is done by computing partial saliency maps for each fixation point (xf, yf) using a 2D Gaussian distribution centered at that point, where these partial maps represent the saliency around each fixation point, and then summing them together to create a saliency map that represents the combined saliency information from all fixation points.

Finally, the global saliency map is normalized by dividing it by its maximum value to ensure that the values are in the range [0, 1].

The formula used to calculate the partial saliency map is:

$$S(I, m) = A \cdot e^{-\frac{(x-x_f)^2+(y-y_f)^2}{2\sigma^2}}$$

where:

S(I, m): The partial saliency map for the image I and fixation point m.

A: A normalization constant.

δ: The Gaussian spread determined by the screen properties and foveal angle.

3 Dataset

The dataset used is the Mexculture142 dataset which is made of images of Mexican Cultural heritage recorded during ANR PI Mexculture by IPN CITEDI and gaze fixations data recorded with an eye-tracker at LABRI UMR 5800 CNRS/University of Bordeaux/IPN. The dataset recorded gaze fixations of subjects executing a visual task of recognition of architectural styles of Mexican Cultural heritage. Each category represents different views of the same architectural structure.

The dataset contains 284 samples having 142 subclasses of Prehispanic, Colonial, Modern buildings, we provide 2 examples for each class and the corresponding .txt files of gaze fixations. Also, the saliency map of each image and .txt scanpath files where we have the coordinates and duration of fixations per subject.

The dataset contains 3 folders:
Images: contains 142 categories of Prehispanic, Colonial and Modern styles.
Fixations: holds .txt files which are the corresponding fixations of source images.
Density Maps: contains the subjective saliency maps of each category.

The identifier for each filename is composed as follows:
Images: SSS_XXX_YYY_N_#.png
Fixations: SSS_XXX_YYY_GazeFix_N_#.txt
Density Maps: SSS_XXX_YYY _GFDM_N_#.png

4 Implementation

The code implementation is made up of multiple script files having a main script file (**main.py**) to be called. The other files include:

1. **constants.py**: which holds all constant values
2. **utils.py**: which holds various utility functions and helper codes used in the project.
    - **calculate_sigma()**:
      This takes in as parameters the image, distance and fovea angle and returns the sigma to be used for on the fixations of each image via equation 2, it then multiplies the equation by an aspect ratio value obtained from comparing screen width & height against image width & height using the condition:

```
if (((SCREEN_WIDTH / image.size[1]) * image.size[0]) ≤ ...
    SCREEN_HEIGHT):
    ratio = image.size[1] / SCREEN_WIDTH
else:
    ratio = image.size[0] / SCREEN_HEIGHT
```

It was however observed and would be discussed later that the results (MSE snd MAE) were better without the use of this ratio.

- **calculate_partial_saliency_map()**:
  This takes in as parameters the image, fixation point as a tuple, and the sigma value to compute and return the partial saliency map as a meshgrid using equation 3.
- **generate_saliency_map()**:
  This uses the image and list of fixation points and obtains; the sigma for the image, the partial saliency maps for all fixations and then the global saliency map by summing all partials and normalizing the result.
- **normalise()**:
  This takes in an array and divides each element by the maximum value to return an array having a range of values between 0 and 1.
- **calculate_error_metrics()**:
  This takes in the GDFM ground truth as well as the obtained saliency map and calculates the following errors for each pair: Mean Absolute error, Mean Squared Error, Pearson's Correlation Coefficient and the Structural similarity index.
- **dict_to_txt()**:
  This takes in a dictionary of images and respective error metrics, creates a txt file and appends the metric values of each image to it.

3. **data_loader.py**: which holds functions for loading images, fixations and GFDM's.
   - **load_image()**:
     It loads an image from its file path and returns a Pillow Image.
   - **load_fixations()**:
     It takes the filepath for the fixation txt file, and returns the fixations as a list of tuples.
   - **load_ground_truth()**:
     This loads the GFDM groundtruth image from its file path and returns a Pillow Image
4. **main.py**
   It is the main python script for this project and it can be called directly from the command line, it has a main function which accepts the folder paths for the images, fixations and groundtruths as arguments, and computes as final outputs the saved global saliency maps, saved blended images of the maps and input image, the error metrics for each image as well as global error metrics for all images and finally saves the metrics in a txt file. The main function is called twice (for both the training images and validation images paths). It can be run by typing the command

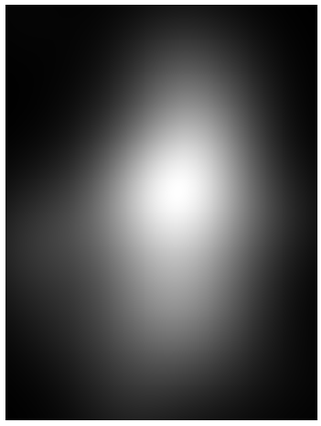
```
python main.py
```

5 Results

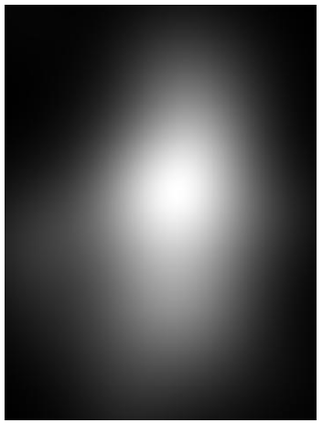
5.1 Visual Evaluation

The results of the project are evaluated using pictures of Saliency Maps and Blended images obtained and are shown below:

5.1.1 Using screen and image width & height ratio



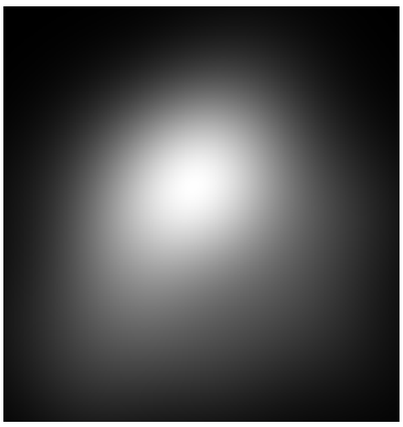
(a) Obtained GFDM



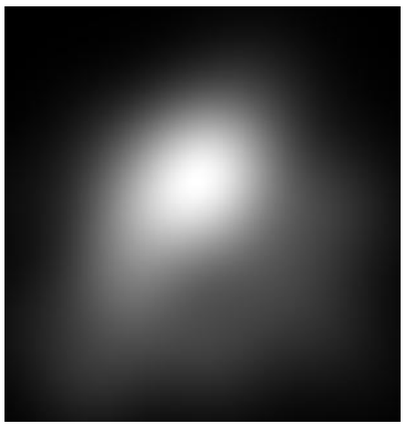
(b) Groundtruth GFDM

(c) Obtained Blended image

Figure 1: Obtained map, Groundtruth and Blended Image



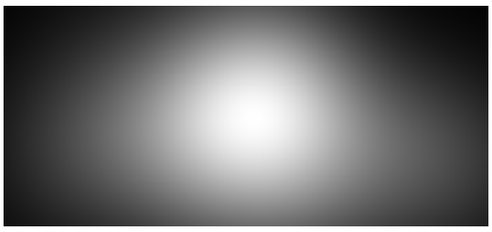
(a) Obtained GFDM



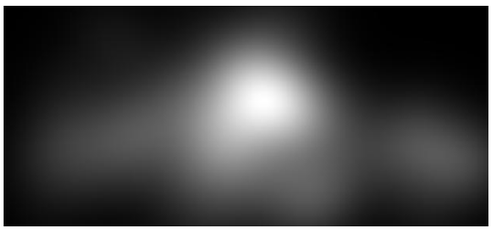
(b) Groundtruth GFDM

(c) Obtained Blended image

Figure 2: Obtained map, Groundtruth and Blended Image



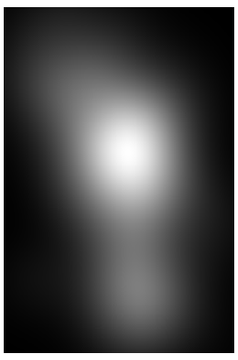
(a) Obtained GFDM



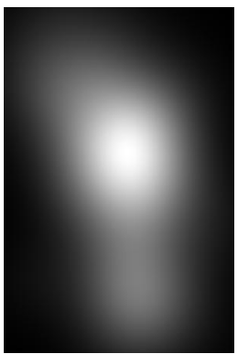
(b) Groundtruth GFDM

(c) Obtained Blended image

Figure 3: Obtained map, Groundtruth and Blended Image



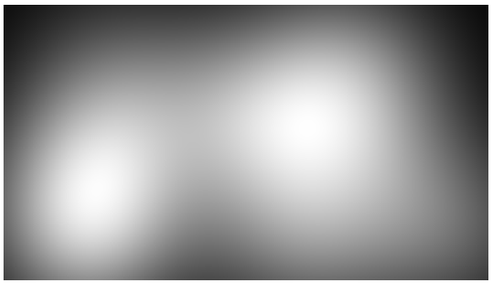
(a) Obtained GFDM



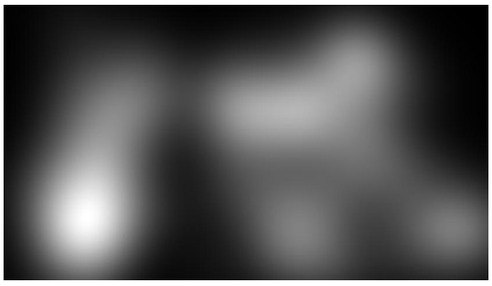
(b) Groundtruth GFDM

(c) Obtained Blended image

Figure 4: Obtained map, Groundtruth and Blended Image



(a) Obtained GFDM

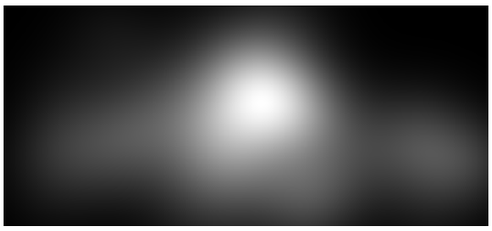


(b) Groundtruth GFDM

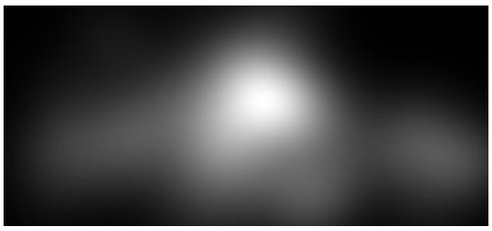
(c) Obtained Blended image

Figure 5: Obtained map, Groundtruth and Blended Image

5.1.2 Without using screen and image width & height ratio



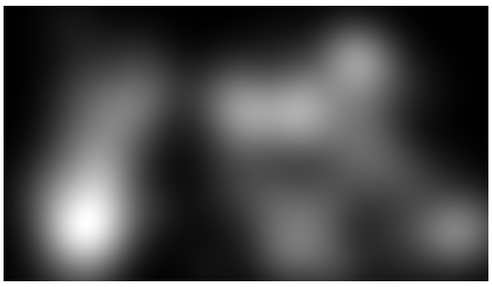
(a) Obtained GFDM



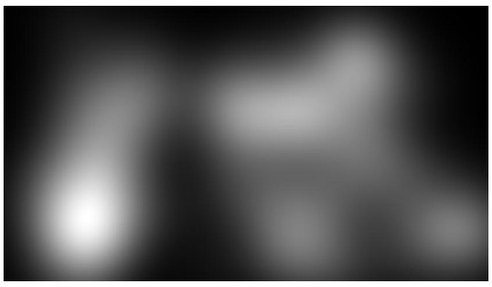
(b) Groundtruth GFDM

(c) Obtained Blended image

Figure 6: Obtained map, Groundtruth and Blended Image



(a) Obtained GFDM



(b) Groundtruth GFDM

(c) Obtained Blended image

Figure 7: Obtained map, Groundtruth and Blended Image

5.2 Quantitative evaluation

The numerical evaluation of the system's performance is done using the following metrics:

1. **Mean Absolute Error**:
   This measures the average absolute pixel-wise difference between the two saliency maps.

2. **Mean Squared Error**:
   This measures the average squared pixel-wise difference between the two saliency maps.

3. **Pearson's Correlation Coefficient**:
   This measures the linear correlation between the pixel values of the two saliency maps. It quantifies the similarity in their intensity distributions. It ranges from -1 (perfect inverse correlation) to 1 (perfect correlation).

4. **Structural Similarity Index**:
   This is a perceptual metric that considers luminance, contrast, and structure information between the two saliency maps. It provides a score between -1 and 1, with 1 indicating a perfect match.

The metrics obtained using all images (train and validation) are presented below:

5.2.1 Using screen and image width & height ratio

Error Metric	Average Value
MAE	0.111988
MSE	0.022597
PCC	0.946029
SSIM	0.747994

Table 1: Average Error Metrics using screen and image width & height ratio

5.2.2 Without using screen and image width & height ratio

Error Metric	Average Value
MAE	0.043916
MSE	0.006760
PCC	0.969239
SSIM	0.886117

Table 2: Average Error Metrics without using screen and image width & height ratio

6 Discussion

Upon analyzing the results obtained in chapter 6, it was observed that the saliency maps obtained without using the ratio performed better in figures 6 and 7 against figures 3 and 5 when the ratios were used, this could be because the resolution of these images have (Width $>$ height).

For the qualitative evaluation, both sets of average results performed well with the results when the ratio wasn't considered having the best performance. Its values as seen in the table are; A very low Mean Absolute Error of 0.043916, A very low Mean Squared Error of 0.006760, a high correlation coefficient of 0.969239 indicating the obtained GFDM pixel values have really close correlation with the Groundtruth GFDM as the value is close to 1 (perfect correlation), and finally a structural similarity index of 0.886117 indicating that the luminance, contrast, and structure information between the two saliency maps are close to each other.

7 Conclusion

In the analysis of Gaze Fixation Density Maps (GFDM) computation based on the MexCulture142 dataset, insights were achieved and the effectiveness of the approach was demonstrated. Here are the key findings and conclusions:

1. **Accuracy and Precision**
   The GFDM computation yielded highly accurate results, as evidenced by the following error metrics:
   Mean Absolute Error (MAE):
   An average MAE of 0.043916 indicates that our computed saliency maps closely resemble the ground truth, with minimal absolute errors between fixation density maps and official maps.
   Mean Squared Error (MSE):
   A low MSE of 0.006760 further supports the precision of our method, signifying that the squared errors between the computed and official maps are consistently low.
   Pearson Correlation Coefficient (PCC):
   With a PCC value of 0.969239, we have established a strong positive correlation between our computed maps and the official maps. This high correlation coefficient validates the accuracy of our GFDM computations.

2. **Structural Similarity** In addition to traditional error metrics, the results were also evaluated using the Structural Similarity Index (SSIM). An SSIM score of 0.886117 indicates that the computed maps exhibit a high degree of structural similarity to the ground truth, demonstrating the effectiveness of the approach in capturing key features of gaze fixation density.