Documentation: calculate_porosity Function

Ramswroop Rathi

June 25, 2023

1 Function Description

The `calculate_porosity` function takes an image path as input, calculates the porosity percentage of the image, and returns the result.

2 Function Signature

```
def calculate_porosity(image_path:  str) -> float
```

3 Parameters

- `image_path` (str): The path to the input image file.

4 Dependencies

The function requires the following dependency:

- `PIL`: Python Imaging Library for image processing.

5 Final answer

The final answer that i got is 31.32%

6 Example Usage

```
from PIL import Image

def calculate_porosity(image_path):
    # Function code goes here

# Example usage
```

```
image_path = 'pros.jpg'
porosity_percentage = calculate_porosity(image_path)
print('Porosity: {:.2f}%'.format(porosity_percentage))
```

7 Algorithm

The `calculate_porosity` function follows the following algorithm:

1. Open the image using the `Image.open` function from the PIL library.
2. Convert the image to grayscale using the `convert` method with the argument `'L'`.
3. Get the pixel data of the grayscale image using the `load` method.
4. Initialize counters for total pixels and pore pixels to 0.
5. Iterate over each pixel in the image using nested `for` loops.
6. Check if the pixel value is below a certain threshold (in this case, 60) to determine if it represents a pore.
7. Increment the corresponding counter based on the result of the pixel check.
8. Calculate the porosity percentage by dividing the number of pore pixels by the total number of pixels and multiplying by 100.
9. Return the porosity percentage.

8 Output

The script outputs the following information:

- Description of each step used by the script and the corresponding output image.
- The final answer of the porosity percentage.

9 Tunable Parameters

The threshold value (60 in this case) used to identify pore pixels can be considered a tunable parameter. If you need to process multiple images with varying characteristics, you can experiment with different threshold values to achieve better accuracy. You can also consider using adaptive thresholding techniques to dynamically determine the threshold based on image properties.

10 Script

Here is the well-commented script in Python:

```
from PIL import Image

def calculate_porosity(image_path):
    # Open the image
    image = Image.open(image_path)

    # Convert the image to grayscale
    grayscale_image = image.convert('L')

    # Get the pixel data
    pixel_data = grayscale_image.load()

    # Initialize counters
    total_pixels = 0
    pore_pixels = 0

    # Iterate over each pixel in the image
    width, height = grayscale_image.size
    for y in range(height):
        for x in range(width):
            # Check if the pixel is dark (representing a pore)
            if pixel_data[x, y] < 60:
                pore_pixels += 1
            total_pixels += 1

    # Calculate the percentage porosity
    porosity = (pore_pixels / total_pixels) * 100

    return porosity

# Provide the path to your image file
image_path = 'pros.jpg'

# Call the function to calculate porosity
porosity_percentage = calculate_porosity(image_path)

# Print the porosity percentage
print('Porosity: {:.2f}%'.format(porosity_percentage))
```