Multithreaded Image Morphology

ECE5772

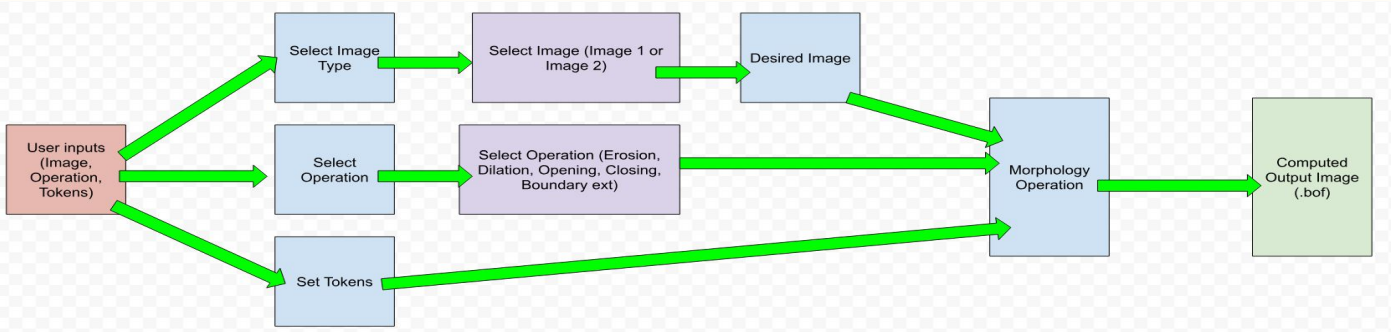
Abdul Wasay

Table of Contents

Project Information

The Purpose of the project is to perform image morphology on images of different sizes and see how implementing parallel_pipeline will affect computation times. Another key aspect of this project was to see how *ntokens* affect parellel_pipeline processing times.

- The program is used via command line
- The two images are read as .bif data
- The output image is saved as a .bof file
- Matlab is used to verify correctness in the output images



Program Inputs

The Program takes 3 user inputs:

- Number of Tokens for Parallel_Pipeline Operation
- Image to Perform operation on
- Morphology Operation to perform on Image

```
./finalproject 100 2 5
```

```
ECE5772 Final Project - Multithreaded Image Morphology:
User Inputs: # of Tokens, Morphology Operation
Images: 1. 640x480  2. 940x602
1. Erosion
2. Dilation
3. Boundary Extraction
4. Opening
5. Closing
```

Program Images

The program allows the user to select between two images: one small and one large

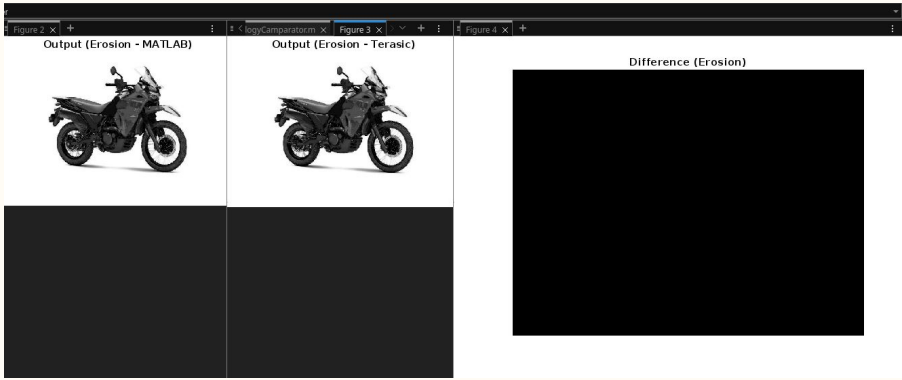
small: 480p

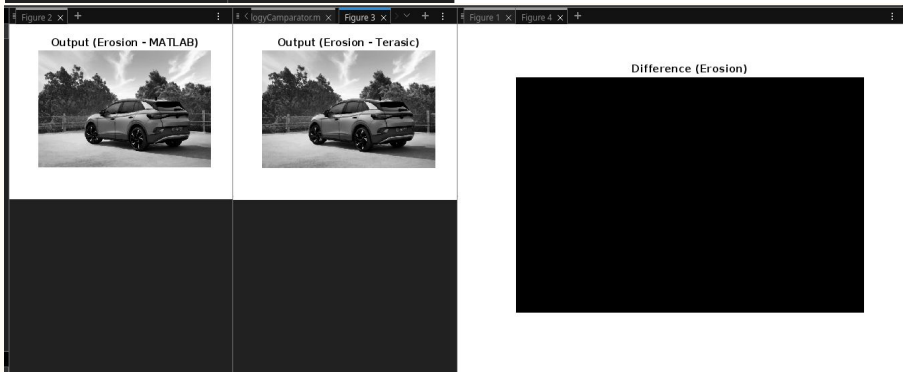
Large: 640p

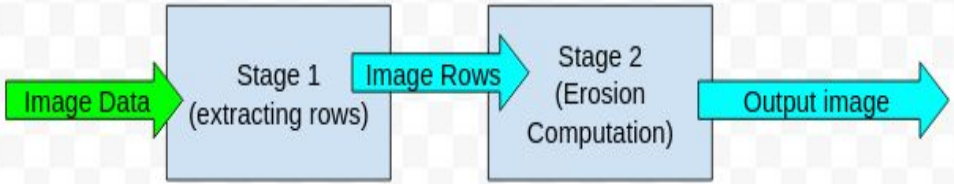
Erosion Operation

Erosion removes pixels from the boundary of images and shrinks them

- Parallel_Pipeline Method consisted of 2 stages
- First stage extracts rows from the input file
- Second Stage performs the Erosion Operation



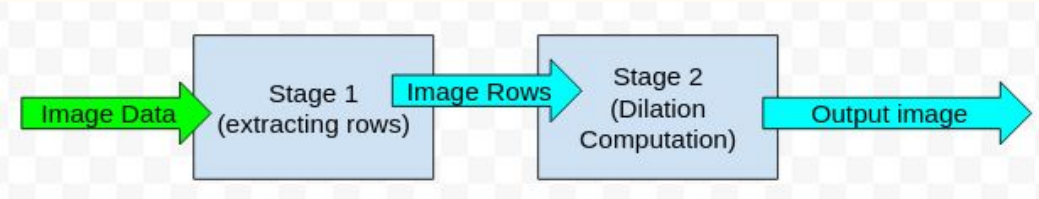


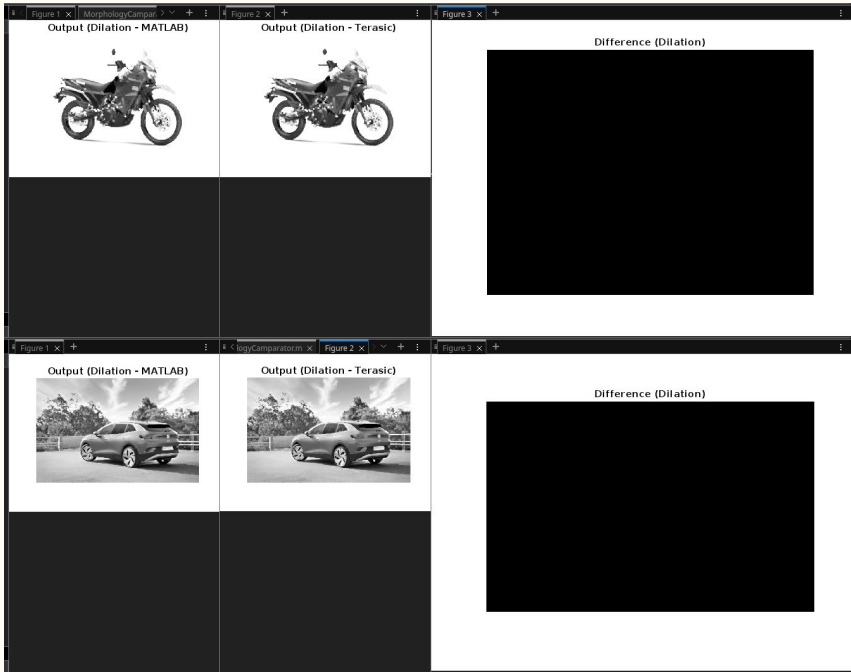


Dilation Operation

Dilation adds pixels to the boundary of objects making them thicker.

- Parallel_Pipeline Method consisted of 2 stages
- First stage extracts rows from the input file
- Second Stage performs the Dilation Operation

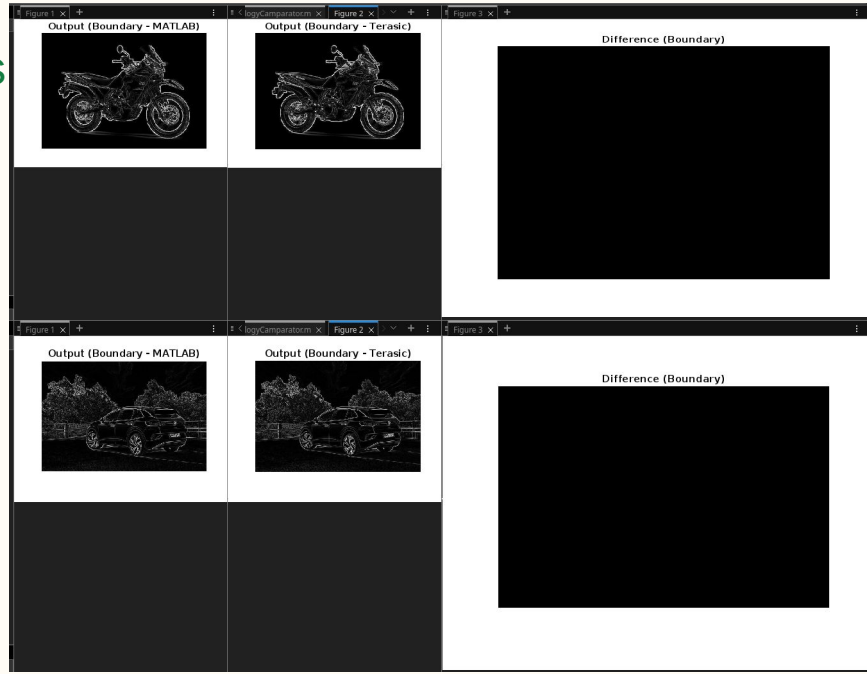


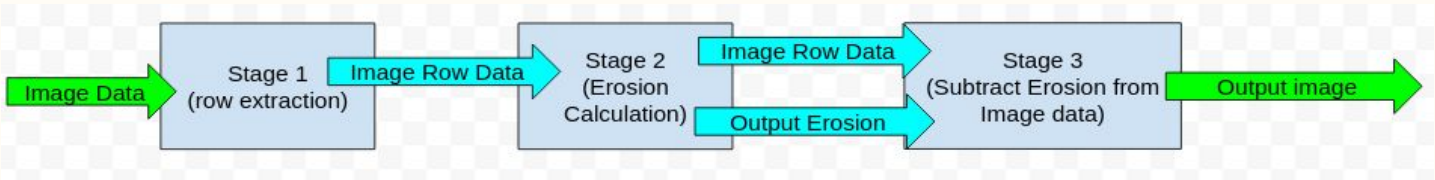


Boundary Extraction Operation

Boundary Extraction outlines the border of images and removes everything else.

- Parallel_Pipeline Method consisted of 3 stages
- First stage extracts rows from the input file
- Second Stage performs the Erosion Operation
- Third Stage subtracts the eroded image from the original

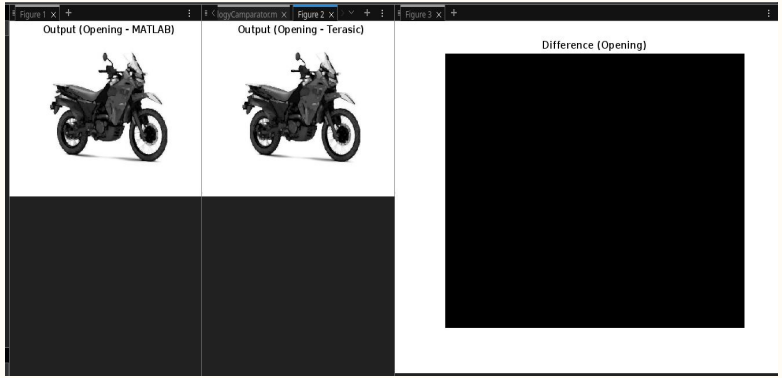


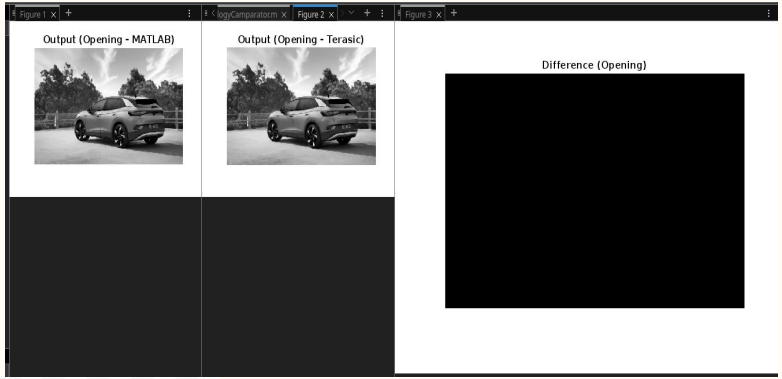


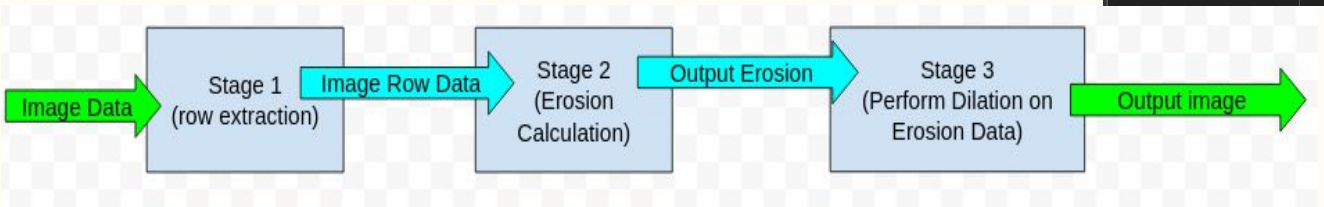
Opening Operation

Opening involves Eroding an image then using that as an input for Dilation.

- Parallel_Pipeline Method consisted of 3 stages
- First stage extracts rows from the input file
- Second Stage performs the Erosion Operation
- Third Stage Uses the Erosion to Perform Dilation



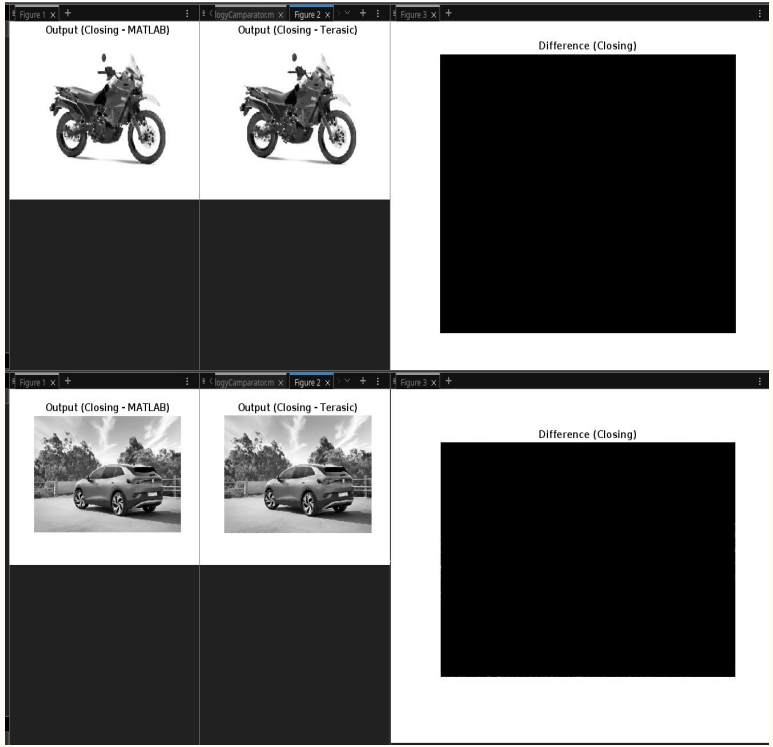


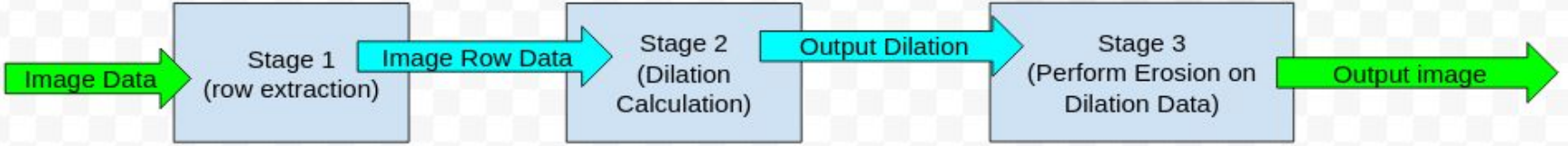


Closing Operation

Closing involves Dilating an image then using that as an input for Erosion.

- Parallel_Pipeline Method consisted of 3 stages
- First stage extracts rows from the input file
- Second Stage performs the Dilation Operation
- Third Stage Uses the Dilation to Perform Erosion





Results

Overall, the program demonstrated that in all Image and Morphology combinations, applying parallelism gave performance gains.

	Image 1						Image 2					
	Ntokens						Ntokens					
	5		16 (Default)		100		5		16 (Default)		100	
Operation	**Sequential**	**Parallel**	**Sequential**	**Parallel**	**Sequential**	**Parallel**	**Sequential**	**Parallel**	**Sequential**	**Parallel**	**Sequential**	**Parallel**
Erosion	21674	13195	16951	8831	19132	9863	59039	21313	56645	20999	62731	23352
Dilation	40736	14294	43971	13294	48494	14680	71980	18834	61802	16440	67541	19030
Boundary Ext	48004	12914	48614	13163	54505	13477	111312	32506	98357	26741	71613	19589
Opening	71440	33220	98426	36267	94601	34101	135073	62975	127115	47665	120369	44757
Closing	90706	36880	80201	30519	86989	32400	135696	56912	143605	54652	135533	51917

Demo

Questions?