Benchmarking

Generated by Doxygen 1.8.11

Contents

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

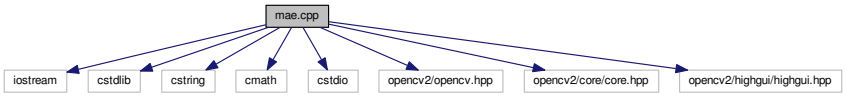
Chapter 2

File Documentation

2.1 mae.cpp File Reference

```
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <cmath>
#include <cstdio>
#include <opencv2/opencv.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
```

Include dependency graph for mae.cpp:



Functions

- static void MAE (cv::Mat img1, cv::Mat img2, int m, int n, int ch)

  Calculates the Mean Absolute Error between the plain image and the encrypted image.
- int main (int argc, char ∗argv[ ])

2.1.1 Function Documentation

2.1.1.1 static void MAE (cv::Mat *img1,* cv::Mat *img2,* int *m,* int *n,* int *ch*) [inline], [static]

Calculates the Mean Absolute Error between the plain image and the encrypted image.

Definition at line 17 of file mae.cpp.

Referenced by main().

```
18 {
19   double init = 0,sum_mae = 0;
20
21   for(int i = 0; i < m; ++i)
22   {
23     for(int j = 0; j < n; ++j)
24     {
25       for(int k = 0; k < ch; ++k)
26       {
27
28         init = (double)img1.at<Vec3b>(i,j)[k] - img2.at<Vec3b>(i,j)[k];
29         sum_mae += fabs(init);
30
31       }
32     }
33   }
34
35   sum_mae = (sum_mae) / (m * n * ch);
36   printf("\nMAE = %F",sum_mae);
37
38 }
```

Here is the caller graph for this function:

2.1.1.2 int main (int *argc*, char * *argv[]*)

Definition at line 40 of file mae.cpp.

References MAE().

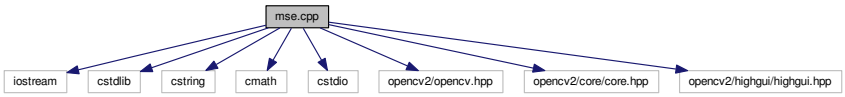
```
41 {
42   std::string image_name_1 = std::string(argv[1]);
43   std::string image_name_2 = std::string(argv[2]);
44
45   cv::Mat img1 = cv::imread(image_name_1,cv::IMREAD_UNCHANGED);
46   cv::Mat img2 = cv::imread(image_name_2,cv::IMREAD_UNCHANGED);
47
48
49
50   int m = img1.rows;
51   int n = img1.cols;
52   int ch = img1.channels();
53
54   cout<<"\nRows =  "<<m;
55   cout<<"\nColumns = "<<n;
56   cout<<"\nChannels = "<<ch;
57   MAE(img1,img2,m,n,ch);
58
59   return 0;
60 }
```

Here is the call graph for this function:

2.2 mse.cpp File Reference

```
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <cmath>
#include <cstdio>
#include <opencv2/opencv.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
```

Include dependency graph for mse.cpp:



Functions

- static void MSE (cv::Mat img1, cv::Mat img2, int m, int n, int ch)

  Calculates the Mean Squared Error between the corresponding plain images of 2 encrypted images, one of which is noisy.
- int main (int argc, char ∗argv[ ])

2.2.1 Function Documentation

2.2.1.1 static void MSE (cv::Mat *img1,* cv::Mat *img2,* int *m,* int *n,* int *ch*) [inline], [static]

Calculates the Mean Squared Error between the corresponding plain images of 2 encrypted images, one of which is noisy.

Definition at line 17 of file mse.cpp.

Referenced by main().

```
18 {
19   double init = 0,sum_mse = 0;
20
21   for(int i = 0; i < m; ++i)
22   {
23     for(int j = 0; j < n; ++j)
24     {
25       for(int k = 0; k < ch; ++k)
26       {
27
28         init = (double)img1.at<Vec3b>(i,j)[k] - img2.at<Vec3b>(i,j)[k];
29         sum_mse += fabs(init);
30
31       }
32     }
33   }
34
35   sum_mse = (sum_mse * 100) / (m * n * ch);
36   printf("\nMSE = %F",sum_mse);
37
38 }
```

Here is the caller graph for this function:

2.2.1.2 int main (int *argc*, char * *argv[]*)

Definition at line 40 of file mse.cpp.

References MSE().

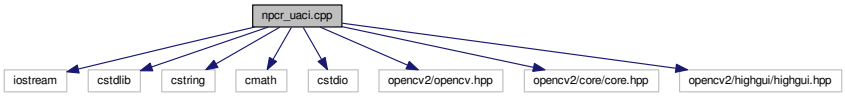
```
41 {
42   std::string image_name_1 = std::string(argv[1]);
43   std::string image_name_2 = std::string(argv[2]);
44
45   cv::Mat img1 = cv::imread(image_name_1,cv::IMREAD_UNCHANGED);
46   cv::Mat img2 = cv::imread(image_name_2,cv::IMREAD_UNCHANGED);
47
48   int m = img1.rows;
49   int n = img1.cols;
50   int ch = img1.channels();
51
52   cout<<"\nRows =  "<<m;
53   cout<<"\nColumns = "<<n;
54   cout<<"\nChannels = "<<ch;
55
56   MSE(img1,img2,m,n,ch);
57
58   return 0;
59 }
```

Here is the call graph for this function:

2.3 npcr_uaci.cpp File Reference

```
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <cmath>
#include <cstdio>
#include <opencv2/opencv.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
```

Include dependency graph for npcr_uaci.cpp:



Functions

- static void NPCR_UACI (cv::Mat img1, cv::Mat img2, int m, int n, int ch)

  Calculates the Number of Pixels Change Rate between two encrypted images whose corresponding plain images differ by a single pixel.
- int main (int argc, char ∗argv[ ])

2.3.1 Function Documentation

2.3.1.1 static void NPCR_UACI (cv::Mat *img1,* cv::Mat *img2,* int *m,* int *n,* int *ch*) [inline], [static]

Calculates the Number of Pixels Change Rate between two encrypted images whose corresponding plain images differ by a single pixel.

Definition at line 17 of file npcr_uaci.cpp.

```
18 {
19   double NPCR = 0,UACI = 0,init = 0,sum = 0,sum_uaci = 0;
20
21   for(int i = 0; i < img1.rows; ++i)
22   {
23     for(int j = 0; j < img1.cols; ++j)
24     {
25       for(int k = 0; k < img1.channels(); ++k)
26       {
27
28         init = (double)img1.at<Vec3b>(i,j)[k] - img2.at<Vec3b>(i,j)[k];
29         sum_uaci += fabs(init);
30         if(img1.at<Vec3b>(i,j)[k] != img2.at<Vec3b>(i,j)[k])
31         {
32           NPCR = 1;
33           sum += NPCR;
34         }
35       }
36     }
37   }
38
39   sum = (sum * 100) / (m * n * ch);
40   printf("\nNPCR = %F",sum);
41   sum_uaci = sum_uaci / (m * n * 255 * ch);
42   printf("\nUACI = %F",sum_uaci * 100);
43 }
```

2.3.1.2 int main (int *argc,* char * *argv[ ]*)

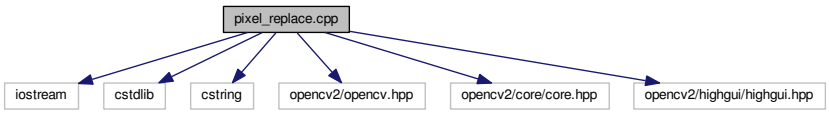
Definition at line 45 of file npcr_uaci.cpp.

```
46 {
47   std::string image_name_1 = std::string(argv[1]);
48   std::string image_name_2 = std::string(argv[2]);
49
50   cv::Mat img1 = cv::imread(image_name_1,cv::IMREAD_UNCHANGED);
51   //cv::Mat img2 = cv::imread(image_name_2,cv::IMREAD_UNCHANGED);
52
53
54
55   int m = img1.rows;
56   int n = img1.cols;
57   int ch = img1.channels();
58
59   cout<<"\nRows =  "<<m;
60   cout<<"\nColumns = "<<n;
61   cout<<"\nChannels = "<<ch;
62
63   //NPCR_UACI(img1,img2,m,n,ch);
64
65   return 0;
66 }
```

2.4 pixel_replace.cpp File Reference

```
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <opencv2/opencv.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
```

Include dependency graph for pixel_replace.cpp:



Functions

- int main (int argc, char **argv)

2.4.1 Function Documentation

2.4.1.1 int main (int *argc,* char ** *argv*)

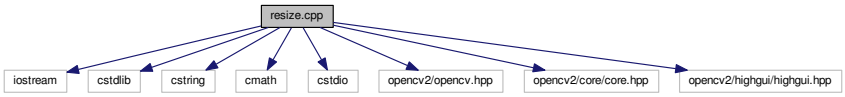
Definition at line 15 of file pixel_replace.cpp.

```
16 {
17
18   std::string name = "lena";
19   std::string image_name = name + ".png";
20   std::string new_image_name = name + "_1pix_" + ".png";
21   cv::Mat image = cv::imread(image_name,cv::IMREAD_UNCHANGED);
22   cout<<"\n"<<image.rows;
23   cout<<"\n"<<image.cols;
24   cout<<"\n"<<image.channels();
25   image.at<Vec3b>(0,0) = 0;
26   cv::imwrite(new_image_name,image);
27   cout<<"\n"<<new_image_name;
28   return 0;
29 }
```

2.5 resize.cpp File Reference

```
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <cmath>
#include <cstdio>
#include <opencv2/opencv.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
```

Include dependency graph for resize.cpp:



Functions

- static std::string getFileNameFromPath (std::string filename)

  Resize an image.
- int main (int argc, char ∗argv[ ])

2.5.1 Function Documentation

2.5.1.1 static std::string getFileNameFromPath (std::string *filename*) [inline],[static]

Resize an image.

Definition at line 15 of file resize.cpp.

Referenced by main().

```
16 {
17   const size_t last_slash_idx = filename.find_last_of("\\/");
18   if (std::string::npos != last_slash_idx)
19   {
20     filename.erase(0, last_slash_idx + 1);
21   }
22
23   // Remove extension if present.
24   const size_t period_idx = filename.rfind('.');
25   if (std::string::npos != period_idx)
26   {
27     filename.erase(period_idx);
28   }
29
30   return filename;
31 }
```

Here is the caller graph for this function:

2.5.1.2 int main (int *argc*, char * *argv[]*)

Definition at line 33 of file resize.cpp.

References getFileNameFromPath().

```
34 {
35   std::string image_path = std::string(argv[1]);
36   std::string image_name = getFileNameFromPath(image_path);
37
38   int rows = atoi((const char*)argv[2]);
39   int cols = atoi((const char*)argv[3]);
40   cv::Mat image = cv::imread(image_path,cv::IMREAD_UNCHANGED);
41   cv::resize(image,image,cv::Size(cols,rows),CV_INTER_LANCZOS4);
42   std::string new_image_name = "";
43   new_image_name  = new_image_name + image_name +  "_" + std::to_string(rows) + "_" + std::to_string(cols)
      + ".png";
44   cout<<"\nNew image name = "<<new_image_name;
45   cv::imwrite(new_image_name,image);
46   return 0;
47 }
```

Here is the call graph for this function:

Index