```
#pragma once

class GetFreq_REC
{
public:

    double* X;  // frequencies buffer pointer
    double** SX_R, ** SX_G, ** SX_B;  //  26x26 pix object RGB components loaded in main program
    double** S, ** S1, ** S2;

    int brUz, sWitch;  // brUz = total nr. of database patterns, sWitch = decides if object is used for db creation or recognition
    double* largest;  // saves probabilities for all db patterns
    double** R;  //  pointer to recognition DB

    double* x;
    fftw_complex* y;
    fftw_plan plan;

    GetFreq_REC();

    //  ***  2d FFTW  ***

    void FFT_REC() {

        int i, j;

        for (i = 0; i < 26; i++)
            for (j = 0; j < 26; j++)
                x[i * 26 + j] = S1[i][j];

        fftw_execute(plan);

```

```
33          for (i = 0; i < 26; i++)
34              for (j = 0; j < 13; j++)
35                  y[i * 14 + j][0] /= 676;
36      }
37
38      void X3_REC() {
39
40          int i, j, k;
41
42          for (i = 0; i < 26; i++) {
43              k = 25;
44              for (j = 0; j < 26; j++) {
45                  S[i][j + 26] = S[i][k--];
46              }
47          }
48          for (i = 0; i < 26; i++)
49              for (j = 0; j < 26; j++) {
50                  S[i][j + 52] = S[i][j];
51              }
52      }
53
54      void Scan_REC() {
55
56          int i, j, m, n;
57          double pom = 0;
58
59          for (i = 0; i < 26; i++) {
60              for (j = 0; j < 13; j++)
61                  S2[i][j] = 0;
62          }
63
64          for (j = 0; j < 52; j++) {
65              for (m = 0; m < 26; m++)
```

```
66                  for (n = j; n < j + 26; n++)
67                      S1[m][n - j] = S[m][n];
68              FFT_REC();
69              for (m = 0; m < 26; m++)
70                  for (n = 0; n < 13; n++)
71                      S2[m][n] += fabs(y[m * 14 + n][0]);
72          }
73      }
74
75      //  ***  probabilities  ***
76
77      void PNN() {
78
79          int i, j;
80          double sum;
81
82          for (i = 0; i < brUz; i++) {
83              sum = 0;
84              for (j = 0; j < 1014; j++)
85                  sum += fabs(X[j] - R[i][j]);
86              largest[i] = exp(-pow(sum, 10) / 2.0E-7);
87          }
88      }
89
90      // ***  main function  ***
91
92      void GetFreq() {
93
94          int i, j, k;
95          long double sum;
96
97          //  *****   Red   *****
98
```

```
 99          for (i = 0; i < 26; i++)
100              for (j = 0; j < 26; j++)
101                  S[i][j] = SX_R[i][j];
102
103          X3_REC();
104          Scan_REC();
105
106          k = 0;
107          for (i = 0; i < 26; i++)
108              for (j = 0; j < 13; j++) {
109                  X[k++] = S2[i][j];
110              }
111
112          //  *****   Green   *****
113
114          for (i = 0; i < 26; i++)
115              for (j = 0; j < 26; j++)
116                  S[i][j] = SX_G[i][j];
117
118          X3_REC();
119          Scan_REC();
120
121          for (i = 0; i < 26; i++)
122              for (j = 0; j < 13; j++) {
123                  X[k++] = S2[i][j];
124              }
125
126          //  *****   Blue   *****
127
128          for (i = 0; i < 26; i++)
129              for (j = 0; j < 26; j++)
130                  S[i][j] = SX_B[i][j];
131
```

```
132         X3_REC();
133         Scan_REC();
134
135         for (i = 0; i < 26; i++)
136             for (j = 0; j < 13; j++) {
137                 X[k++] = S2[i][j];
138             }
139
140         //  ***  normalize & log ***
141
142         sum = 0;;
143
144         for (i = 0; i < 1014; i++)
145             sum += X[i];
146         sum /= 1014;
147         if (sum > 0) {
148             for (i = 0; i < 1014; i++)
149                 X[i] /= sum;
150
151             for (i = 0; i < 1014; i++)
152                 X[i] = log(X[i] + 1);
153         }
154
155         sum = 0;
156         for (i = 0; i < 1014; i++)
157             sum += X[i];
158         if (sum > 0)
159             for (i = 0; i < 1014; i++)
160                 X[i] /= sum;
161
162         if (sWitch == 1) PNN();
163     }
164
```

```
165     ~GetFreq_REC();
166 };
167
168
```