```
1
2
3       typedef double XFLOAT
4       typedef double OTA_FLOAT
5
6
7   {
8
9       XFLOAT CalculateNoiseIndicator_SpeechLQ(const PolqaStatics *statics, CPOLQAData *POLQAHandle,
    const  CBarkSpectrum& originalPitchPowerDensity_intact, const CBarkSpectrum&
    distortedPitchPowerDensity_intact, const int* ActiveRatioOkFlags,
10          const int numberOfSilentFrames, const int numberOfNotSilentFrames, CIntArray& ActiveRatioOk,
    const CIntArray& SilentFrameFlags, CDoubleArray* AddedSilentDisturbance, XFLOAT*
    noiseIndicatorTimbre, XFLOAT* noiseIndicatorTimbreAdd,
11          XFLOAT* noiseIndicatorFreqImpact, XFLOAT* noiseIndicatorScalingImpact, XFLOAT*
    delayJumpCompNB, XFLOAT* delayJumpCompWB, CIntArray& aActive,
12          CBarkSpectrum& originalLoudnessDensity, CBarkSpectrum& distortedLoudnessDensity,
    CBarkSpectrum& originalPitchLoudnessDensityMainAvg, CBarkSpectrum&
    distortedPitchLoudnessDensityMainAvg, XFLOAT* noiseIndicatorHighBandsCompensation000)
13      {
14
15          CBarkSpectrum     originalPitchPowerDensitySilent, distortedPitchPowerDensitySilent
16          CBarkSpectrum     originalLoudnessDensitySilent, distortedLoudnessDensitySilent
17          CBarkSpectrum     disturbanceDensityAddSilent
18          CTimeSeries       aNoise
19
20          CNewStdString s
21
22          int frameIndex
23          int numberOfSpeechFrames = statics->stopFrameIdx - statics->startFrameIdx
24          originalPitchPowerDensitySilent.Initialize("originalPitchPowerDensitySilent", POLQAHandle)
25          distortedPitchPowerDensitySilent.Initialize("distortedPitchPowerDensitySilent", POLQAHandle)
26          originalLoudnessDensitySilent.Initialize("originalLoudnessDensitySilent", POLQAHandle)
27          distortedLoudnessDensitySilent.Initialize("distortedLoudnessDensitySilent", POLQAHandle)
28          disturbanceDensityAddSilent.Initialize("disturbanceDensityAddSilent", POLQAHandle)
29
30          originalPitchPowerDensitySilent.FrequencyWarpingOf(POLQAHandle, originalHzPowerSpectrum, 1.0)
31          distortedPitchPowerDensitySilent.FrequencyWarpingOf(POLQAHandle, distortedHzPowerSpectrum,
    PitchRatio)
32
33          originalPitchPowerDensityPureFrq.FrequencyWarpingOf(POLQAHandle, originalHzPowerSpectrum,
    1.0)
34          distortedPitchPowerDensityPureFrq.FrequencyWarpingOf(POLQAHandle, distortedHzPowerSpectrum,
    PitchRatio)
35
36          XFLOAT hulp
37          XFLOAT hulp1 = 0.0
38          XFLOAT hulp2 = 0.0
39
40          for (frameIndex = statics->startFrameIdx  frameIndex <= statics->stopFrameIdx  frameIndex++)
41          {
42              if (ActiveRatioOkFlags[frameIndex])
43              {
44                  hulp1 += originalPitchPowerDensitySilent.TotalAudible(POLQAHandle, frameIndex, 1.0)
45                  hulp2 += distortedPitchPowerDensitySilent.TotalAudible(POLQAHandle, frameIndex, 1.0)
46              }
47          }
48
49          hulp1 /= (numberOfSpeechFrames + 0.01)
50          hulp2 /= (numberOfSpeechFrames + 0.01)
51          hulp1 = 1.0e6 / (hulp1 + 1.0)
52          hulp2 = 1.0e6 / (hulp2 + 1.0)
53
54          for (frameIndex = statics->startFrameIdx  frameIndex <= statics->stopFrameIdx  frameIndex++)
55          {
56              originalPitchPowerDensitySilent.MultiplyWith(frameIndex, hulp1)
57              distortedPitchPowerDensitySilent.MultiplyWith(frameIndex, hulp2)
58          }
```

```
59
60          for (frameIndex = statics->startFrameIdx  frameIndex <= statics->stopFrameIdx  frameIndex++)
61          {
62              if (originalPitchPowerDensitySilent.TotalAudible(POLQAHandle, frameIndex, 1.0) < 3.0E6)
63                  aActive.m_pData[frameIndex] = FALSE  else
64                  aActive.m_pData[frameIndex] = TRUE
65          }
66
67          originalLoudnessDensitySilent.IntensityWarpingOf(POLQAHandle,
    originalPitchPowerDensitySilent)
68          distortedLoudnessDensitySilent.IntensityWarpingOf(POLQAHandle,
    distortedPitchPowerDensitySilent)
69
70          originalPitchLoudnessDensityMainAvg.TimeLpOf(POLQAHandle, originalLoudnessDensitySilent,
    ActiveRatioOk, 1.0)
71          distortedPitchLoudnessDensityMainAvg.TimeLpOf(POLQAHandle, distortedLoudnessDensitySilent,
    ActiveRatioOk, 1.0)
72          originalLoudnessDensitySilent.AudibleFreqRespCompensationExact(originalPitchLoudnessDensityMa
    inAvg,
73              distortedPitchLoudnessDensityMainAvg, 0.1)
74
75          disturbanceDensityAddSilent.DifferenceOfBandlimited(distortedLoudnessDensitySilent,
    originalLoudnessDensitySilent)
76
77          disturbanceDensityAddSilent.Orthogonalize(aActive)
78
79          disturbanceDensityAddSilent.ComputeLpWeights(POLQAHandle, MINIMUM_POWER_FREQ,
    STEP_POWER_FREQ, NUMBER_OF_POWERS_OVER_FREQ, AddedSilentDisturbance)
80
81          XFLOAT noiseIndicatorAlignJumps, noiseIndicatorAlignJumpsMax
82          XFLOAT noiseIndicatorHighBands, noiseIndicatorPulsImpact, signalLoudness,
    signalLoudnessDistHighBands
83
84          XFLOAT noiseIndicator = 0.0
85          signalLoudness = 0.0
86          noiseIndicatorHighBands = 0.0
87          signalLoudnessDistHighBands = 0.0
88          for (frameIndex = statics->startFrameIdx  frameIndex <= statics->stopFrameIdx  frameIndex++)
89          {
90              if (SilentFrameFlags[frameIndex])
91              {
92                  noiseIndicator += disturbanceDensityAddSilent.Total(frameIndex, 300.0, 3500.0)
93                  noiseIndicatorHighBands += disturbanceDensityAddSilent.Total(frameIndex, 3200.0,
    3.0e4)
94              }
95              else {
96                  signalLoudness += originalLoudnessDensity.Total(frameIndex, 0.0, 3.0e4)
97                  signalLoudnessDistHighBands += distortedLoudnessDensity.Total(frameIndex, 3400.0,
    3.0e4)
98              }
99          }
100         if (noiseIndicator < 0.0) noiseIndicator = 0.0
101         noiseIndicator /= (numberOfSilentFrames + 0.01)
102
103         noiseIndicatorPulsImpact = noiseIndicator
104         if (noiseIndicatorPulsImpact < 1.0) noiseIndicatorPulsImpact = 1.0
105         if (noiseIndicatorPulsImpact > 10.0) noiseIndicatorPulsImpact = 10.0
106         noiseIndicatorPulsImpact = pow(noiseIndicatorPulsImpact, 0.1)
107
108         *noiseIndicatorFreqImpact = noiseIndicator
109         if (*noiseIndicatorFreqImpact < 1.0) *noiseIndicatorFreqImpact = 1.0
110         *noiseIndicatorFreqImpact = pow(*noiseIndicatorFreqImpact, 0.1)
111
112         *noiseIndicatorScalingImpact = noiseIndicator - 0.3
113         if (*noiseIndicatorScalingImpact < 0.0) *noiseIndicatorScalingImpact = 0.0
114         if (*noiseIndicatorScalingImpact > 20.0) *noiseIndicatorScalingImpact = 20.0
115         *noiseIndicatorScalingImpact = pow(*noiseIndicatorScalingImpact, 0.55)
116
```

```
        if (noiseIndicatorHighBands < 0.0) noiseIndicatorHighBands = 0.0
        noiseIndicatorHighBands /= (numberOfSilentFrames + 0.01)
        if (noiseIndicatorHighBands > 1.8) noiseIndicatorHighBands = 1.8
        signalLoudnessDistHighBands /= (numberOfNotSilentFrames + 0.01)
        hulp = signalLoudnessDistHighBands - noiseIndicatorHighBands
        if (hulp < 9.0) hulp = 9.0
        *noiseIndicatorHighBandsCompensation000 = 0.7*noiseIndicatorHighBands / hulp
        s.Format("noiseIndicatorHighBandsCompensation000=%f  noiseIndicatorHighBands=%f
signalLoudnessDistHighBands=%f  \n ", noiseIndicatorHighBandsCompensation000,
noiseIndicatorHighBands, signalLoudnessDistHighBands)


        noiseIndicatorAlignJumpsMax = 300.0
        noiseIndicatorAlignJumps = (noiseIndicator - 77.0)
        if (noiseIndicatorAlignJumps < 1.0) noiseIndicatorAlignJumps = 1.0
        if (noiseIndicatorAlignJumps > noiseIndicatorAlignJumpsMax) noiseIndicatorAlignJumps =
noiseIndicatorAlignJumpsMax

        *delayJumpCompWB = 19.0*pow(noiseIndicatorAlignJumps, 0.04)
        *delayJumpCompNB = 80.0

        *noiseIndicatorTimbre = (noiseIndicator - 73.0)
        if (*noiseIndicatorTimbre < 1.0) *noiseIndicatorTimbre = 1.0
        *noiseIndicatorTimbreAdd = (noiseIndicator - 8.0)
        if (*noiseIndicatorTimbreAdd < 1.0) *noiseIndicatorTimbreAdd = 1.0
        if (*noiseIndicatorTimbre > noiseIndicatorAlignJumpsMax) *noiseIndicatorTimbre =
noiseIndicatorAlignJumpsMax
        if (*noiseIndicatorTimbreAdd > noiseIndicatorAlignJumpsMax) *noiseIndicatorTimbreAdd =
noiseIndicatorAlignJumpsMax
        *noiseIndicatorTimbre = pow(*noiseIndicatorTimbre, 0.2)
        *noiseIndicatorTimbreAdd = pow(*noiseIndicatorTimbreAdd, 0.3) - 0.3

        signalLoudness /= ((numberOfSpeechFrames - numberOfSilentFrames) + 0.01)
        noiseIndicator /= (signalLoudness + 0.01)
        if (noiseIndicator < 0.2) noiseIndicator = 0.2
        noiseIndicator = pow(noiseIndicator, 0.1)
        return noiseIndicator
    }
}
```