Implementer's Guide II for Recommendation ITU-T P.863: Bug fixes

1. Summary

During the characterization phase of P.863 some bugs were discovered in the code used to implement the temporal alignment of the algorithm. This corrigendum contains a list of the discovered programming errors and fixes for those bugs. The influence of the fixes on the resulting performance is presented and updated conformance test results are also provided.

2. List of Bugs

The list of bugs in this paragraph uses a notation which should be well understandable by everybody familiar with the C++ programming language (and others are probably not interested in this level of detail…).

```
------------------------------------------------------------
#define TABUG_1
SpeechTempAlign.cpp, line 227,
CSpeechDelaySearch::CombineMatricesAndFeatures():

Arguments of matbCopy() are swapped.

#ifdef TABUG_1
     // Proposed change:
     matbCopy(ppMatrix[Deg], ppMatrix[Start], DelayFrames);
#else
     // Original code version:
     matbCopy(ppMatrix[Start], ppMatrix[Deg], DelayFrames);
#endif

------------------------------------------------------------
#define TABUG_2
SpeechTempAlign.cpp, lines 302-304,
CSpeechDelaySearch::CleanupPath()

Detection of delay changes which are reverted after a short period does not take
sign of the delay reversal into account.

#ifdef TABUG_2
     // Proposed change:
     bool ChangeIsPositive = ((int)(DelayVec[i] - DelayVec[i-1])) > 0;
     for (k=i+1; k<DelayVecLen && !Found; k++)
     {
          int NextChange = (int)(DelayVec[k] - DelayVec[k-1]);
          if (abs(NextChange) > LargeChange2 && ((NextChange > 0) !=
                         ChangeIsPositive))
               Found=true;
     }
#else
     // Original code version:
     for (k=i+1; k<DelayVecLen && !Found; k++)
          if (abs((int)(DelayVec[k] - DelayVec[k-1])) > LargeChange2)
               Found=true;
#endif
```

```
------------------------------------------------------------
#define TABUG_3
TempAlignment.cpp, lines 3344/3345
CTempAlignment::CoarseAlignment()

DelayVec is erratically overwritten with last valid delays after
CombineMatricesAndFeatures(). This reverts some previous steps and is not
intended.

#ifdef TABUG_3
     // Proposed change:
#else
     // Original code version:
     for (d=0; d<mNumMacroFrames; d++)
          DelayVec[d] = DelayVec[FrameWithLastValidDelay[d]];
#endif


------------------------------------------------------------
#define TABUG_4
TempAlignment.cpp, line 3655
CTempAlignment::Run()

Variable "DelayAfter" is used in units of feature frames instead of samples.

#ifdef TABUG_4
     // Proposed change:
     ChangePosDeg = max(0, RefEndSample + (int)(0.5*(RefStartSample-
                    RefEndSample)) - DelayAfter * IterationData.mStepSize);
#else
     // Original code version:
     ChangePosDeg = RefEndSample + (int)(0.5*(RefStartSample-RefEndSample)) –
                    DelayAfter;
#endif


------------------------------------------------------------
#define TABUG_6
SpeechTempAlign.cpp, lines 276
CSpeechDelaySearch::CleanupPath()

Index k is off by 1 due to loop counter.

#ifdef TABUG_6
     // Proposed change:
     k--;
#endif
```

```
------------------------------------------------------------
#define TABUG_7
SpeechTempAlign.cpp, line 249
CSpeechDelaySearch::CombineMatricesAndFeatures()

Like TABUG_1, arguments swapped

#ifdef TABUG_7
     // Proposed change:
     matbCopy(ppMatrix[StartFrame], ppMatrix[i], DelayFrames);
#else
     // Original code version:
     matbCopy(ppMatrix[i], ppMatrix[StartFrame], DelayFrames);
#endif



------------------------------------------------------------
#define TABUG_8
   - May require retraining of MinPauseLength2!
   - No difference on POLQA set.
   - Code not part of public C code.

SpeechActiveFrameDetection.cpp
CSpeechActiveFrameDetection::IdentifyActivity()

Missing check for condition in VAD of OPTICOM prealignment.

#ifdef TABUG_8
     // Proposed change:
     if(MinEnergyExceeded)
          for( iteration = finish; iteration < start; iteration++ )
               Vec[iteration] = LevelMin;
#else
     // Original code version:
     for( iteration = finish; iteration < start; iteration++ )
          Vec[iteration] = LevelMin;
#endif



------------------------------------------------------------
#define TABUG_9
Idealization.cpp, lines 362
CPairParameters::IdealizationProcess()

Potential buffer overrun by incorrect check for limits.

#ifdef TABUG_9
     // Proposed change:
     int StopFrameDeg = min(mMaxModelFrames-1, min(n-1, (StopSampleDeg /
          (aTransformLength/2)) - 1));
#else
     // Original code version:
     int StopFrameDeg = min(mMaxModelFrames - 1, (StopSampleDeg /
          (aTransformLength/2)) - 1);
```

```
#endif

------------------------------------------------------------
#define TABUG_10
OptInterface.h, line 93
CPairParameters::IdealizationProcess()

Lower threshold for resampling


#ifdef TABUG_10
	#define USE_SRC_THRESHOLD	0.005
#else
	#define USE_SRC_THRESHOLD	0.01
#endif
```

3. Influence of the Bug Fixes on the Conformance Test Results

Updated conformance test results are provided by the attached zip archive. Those files should replace the ones distributed with ITU-T P.863.

4. Update to Appendix I of ITU-T P.863

The application of the above bug fixes changes the results of P.863 slightly. The results presented in Appendix I of P.863 have to be updated accordingly. A new version of this appendix is found below. Please note that all data were produced including the mapping function for P.863, see Implementer's Guide I to P.863.

I.3 Performance results for the ITU-T P.863 algorithm

The two rmse* values for the ITU-T P.863 algorithm per database shown in Tables I.1 to I.3 are obtained after a monotonous 3rd order mapping or after a 1st order linear mapping of the ITU-T P.863 results, respectively. The rmse* values reflect the 'per condition' performance of the ITU-T P.863 algorithm. The prediction becomes less accurate as the rmse* increases. Values of rmse* < 0.1 can be seen as proper and accurate predictions. The prediction becomes less accurate as the rmse* increases; however, high values can be caused by either single outliers or a general lower prediction accuracy. The ITU-T P.863 algorithm does not result for any tested database in a rmse* > 0.3.

Table I.1 – Two rmse* values for ITU-T P.863 per database (Set 1)

Database	**rmse* 3rd**	**rmse* 1st**
Set 1	**ITU-T P.863**	**ITU-T P.863**
NB_BT_P862_BGN_ENG	0.1027	0.1928
NB_BT_P862_PROP	0.1654	0.1672
NB_DT_P862_1st	0.1476	0.1668

NB_DT_P862_BGN_GER	0.1110	0.1529
NB_DT_P862_Share	0.0903	0.0912
NB_ERIC_AMR_4B	0.1352	0.1410
NB_ERIC_P862_NW_MEAS	0.1614	0.1635
NB_TNO_P862_KPN_KIT97	0.2371	0.2586
NB_TNO_P862_NW_EMU	0.1489	0.1503
NB_TNO_P862_NW_MEAS	0.1764	0.1887
NB_ITU_SUPPL23_EXP1a	0.1014	0.1116
NB_ITU_SUPPL23_EXP1d	0.0665	0.0663
NB_ITU_SUPPL23_EXP1o	0.1030	0.1127
NB_ITU_SUPPL23_EXP3a	0.1705	0.2222
NB_ITU_SUPPL23_EXP3c	0.0905	0.1331
NB_ITU_SUPPL23_EXP3d	0.0553	0.0681
NB_ITU_SUPPL23_EXP3o	0.0585	0.0887
NB_FT_P563_PROP	0.0632	0.0669
NB_LUC_P563_PROP	0.0982	0.1118
NB_OPT_P563_PROP	0.1186	0.1242
NB_PSY_P563_PROP	0.1763	0.2070
NB_SQ_P563_PROP	0.1722	0.1749
Average	0.1250	0.1437

Table I.2 – Two rmse* values for ITU-T P.863 per database (Set 2)

Database	**rmse* 3rd**	**rmse* 1st**
Set 2	**ITU-T P.863**	**ITU-T P.863**
NB_ATT_iLB	0.1951	0.2141
NB_ERIC_Field_GSM_EU	0.1578	0.1717
NB_ERIC_Field_GSM_US	0.1475	0.1490
NB_GIPS_EXP1	0.0946	0.1550
WB_GIPS_EXP3	0.1291	0.1728
SWB_GIPS_EXP4	0.0776	0.0771
NB_QUALCOMM_EXP1b	0.1090	0.1246
WB_QUALCOMM_EXP1w	0.1194	0.1468
NB_QUALCOMM_EXP2b	0.1462	0.1461

NB_QUALCOMM_EXP3w	0.0865	0.0963
WB_QUALCOMM_EXP3w	0.0713	0.1457
SWB_48kHz101_ERICSSON	0.2808	0.2984
WB_48kHz102_ERICSSON	0.1925	0.1959
SWB_48kHz201_FT_DT	0.2686	0.2732
SWB_48kHz202_FT_DT	0.2458	0.2519
SWB_48kHz301_OPTICOM	0.2780	0.3015
SWB_48kHz302_OPTICOM	0.1867	0.2125
SWB_48kHz401_PSYTECHNICS	0.1491	0.1690
WB_48kHz402_PSYTECHNICS	0.1812	0.1775
SWB_48kHz501_SWISSQUAL	0.2040	0.2262
SWB_48kHz502_SWISSQUAL	0.2618	0.2634
SWB_48kHz601_TNO	0.2200	0.2184
SWB_48kHz602_TNO	0.1717	0.1952
Average	0.1728	0.1905

Table I.3 – Two rmse* values for ITU-T P.863 per database (Set 3)

Database	**rmse* 3rd**	**rmse* 1st**
Set 3	**ITU-T P.863**	**ITU-T P.863**
SWB_48kHz103_ERICSSON	0.2277	0.2473
NB_8kHz104_ERICSSON	0.2846	0.2824
SWB_48kHz203_FT_DT	0.2809	0.2790
WB_16kHz204_FT_DT	0.2298	0.2297
SWB_48kHz303_OPTICOM	0.1819	0.1987
SWB_48kHz403_PSYTECHNICS	0.1756	0.1766
NB_48kHz404_PSYTECHNICS	0.1814	0.1799
SWB_48kHz503_SWISSQUAL	0.2085	0.2076
NB_8kHz504_SWISSQUAL	0.2276	0.2302
SWB_48kHz603_TNO	0.1550	0.1610
NB_8kHz_NTT_PTEST_1	0.0879	0.0881
NB_QUALCOMM_EXP4	0.1209	0.1310
WB_QUALCOMM_EXP5	0.1060	0.1612
NB_QUALCOMM_EXP6a	0.2147	0.2241

NB_QUALCOMM_EXP6b	0.1176	0.1802
NB_16kHz_HUAWEI_1	0.1289	0.1344
NB_16kHz_HUAWEI_2	0.1908	0.2293
Average	0.1835	0.1965

An important parameter is the worst case performance that was also part of the evaluation procedure.

Table I.4 – Worst case performance of the ITU-T P.863 algorithm (Sets 1 to 3)

Absolute worst case over the three sets	0.2846	0.3015
Average of the three worst experiments	0.2821	0.2941

The performance of the ITU-T P.863 algorithm compared to the algorithm [b-ITU-T P.862] is shown in Tables I.5 and I.6. Here the ITU-T P.862 values are mapped using [b-ITU-T P.862.1]. Tables I.5 and I.6 are restricted to narrow-band databases. The ITU-T P.863 algorithm shows a reduction of rmse* by 27% compared with that of [b-ITU-T P.862.1] after 3rd order mapping and one of 30% after first order mapping.

Table I.5 – Performance of ITU-T P.863 compared to ITU-T P.862 (Set A)

Database	**rmse* 3rd**		**rmse* 1st**	
Set A (narrowband)	**ITU-T P.862.1**	**ITU-T P.863**	**ITU-T P.862.1**	**ITU-T P.863**
NB_BT_P862_BGN_ENG	0.149	0.1027	0.2182	0.1269
NB_BT_P862_PROP	0.1603	0.1654	0.1860	0.1684
NB_DT_P862_1st	0.1916	0.1476	0.207	0.1830
NB_DT_P862_BGN_GER	0.0973	0.1110	0.1465	0.1132
NB_DT_P862_Share	0.1263	0.0903	0.1276	0.0893
NB_ERIC_AMR_4B	0.0918	0.1352	0.0999	0.1573
NB_ERIC_P862_NW_MEAS	0.2214	0.1614	0.2406	0.1759
NB_TNO_P862_KPN_KIT97	0.2967	0.2371	0.3370	0.2089
NB_TNO_P862_NW_EMU	0.3017	0.1489	0.2983	0.1567
NB_TNO_P862_ NW_MEAS	0.2493	0.1764	0.2654	0.1724
NB_ITU_SUPPL23_EXP1a	0.1342	0.1014	0.1644	0.1195
NB_ITU_SUPPL23_EXP1d	0.078	0.0665	0.0957	0.0661
NB_ITU_SUPPL23_EXP1o	0.1091	0.1030	0.1386	0.1232
NB_ITU_SUPPL23_EXP3a	0.1939	0.1705	0.2357	0.2035

NB_ITU_SUPPL23_EXP3c	0.137	0.0905	0.1891	0.0989
NB_ITU_SUPPL23_EXP3d	0.1258	0.0553	0.1290	0.0646
NB_ITU_SUPPL23_EXP3o	0.1537	0.0585	0.1725	0.0623
NB_FT_P563_PROP	0.1139	0.0632	0.1188	0.0653
NB_LUC_P563_PROP	0.0632	0.0982	0.0821	0.1234
NB_OPT_P563_PROP	0.115	0.1186	0.1315	0.1179
NB_PSY_P563_PROP	0.1623	0.1763	0.1696	0.1848
NB_SQ_P563_PROP	0.1915	0.1722	0.1941	0.1815
Average	0.1574	0.1250	0.1795	0.1347

Table I.6 – Performance of ITU-T P.863 compared to ITU-T P.862 (Set B)

Database	**rmse* 3rd**		**rmse* 1st**	
Set B (narrowband)	**ITU-T P.862.1**	**ITU-T P.863**	**ITU-T P.862.1**	**ITU-T P.863**
NB_ATT_iLBC	0.2268	0.1951	0.2243	0.2305
NB_ERIC_Field_GSM_EU	0.2401	0.1578	0.2458	0.1548
NB_ERIC_Field_GSM_US	0.1986	0.1475	0.2245	0.1491
NB_GIPS_EXP1	0.2943	0.0946	0.3906	0.1257
NB_QUALCOMM_EXP1b	0.1588	0.109	0.2593	0.1232
NB_QUALCOMM_EXP2b	0.1826	0.1462	0.2591	0.1489
NB_QUALCOMM_EXP3w	0.1546	0.0865	0.2219	0.0972
NB_8kHz104_ERICSSON	0.357	0.2846	0.3826	0.2788
NB_48kHz404_PSYTECHNICS	0.326	0.1814	0.3412	0.1661
NB_8kHz504_SWISSQUAL	0.4203	0.2276	0.4166	0.2355
NB_8kHz_NTT_PTEST_1	0.1073	0.0879	0.1150	0.0916
NB_QUALCOMM_EXP4	0.173	0.1209	0.2533	0.1265
NB_QUALCOMM_EXP6a	0.248	0.2147	0.3074	0.2130
NB_QUALCOMM_EXP6b	0.1491	0.1176	0.2865	0.1373
NB_16kHz_HUAWEI_1	0.1719	0.1289	0.2026	0.1288
Average	0.2272	0.1534	0.2754	0.1605

Table I.7 gives the performance of the ITU-T P.863 algorithm compared with that of [b-ITU-T P.862.2]. In the ITU-T P.863 evaluation set six 'common' wideband databases (up to 7'000 Hz audio bandwidth) were used. The ITU-T P.863 algorithm results in a reduced rmse* of 56% compared with that of [b-ITU-T P.862.2].

Table I.7 – Performance of ITU-T P.863 compared to ITU-T P.862.2 (Set C)

Database	rmse* 3rd		rmse* 1st	
Set C (wideband)	**ITU-T P.862.2**	**ITU-T P.863**	**ITU-T P.862.2**	**ITU-T P.863**
WB_48kHz102_ERICSSON	0.4521	0.1925	0.4482	0.1920
WB_16kHz402_PSYTECHNICS	0.3245	0.1812	0.3646	0.1838
WB_GIPS_EXP3	0.3467	0.1291	0.4255	0.1530
WB_QUALCOMM_EXP1w	0.2606	0.1194	0.3664	0.1384
WB_QUALCOMM_EXP3w	0.2852	0.0713	0.3727	0.1099
WB_16kHz204_FT_DT	0.4221	0.2298	0.4158	0.2335
WB_QUALCOMM_EXP5	0.3236	0.106	0.3773	0.1413
Average	0.345	0.1470	0.3958	0.1646