Application Note

Document Number

Measure Turbidity with Spectral Sensors

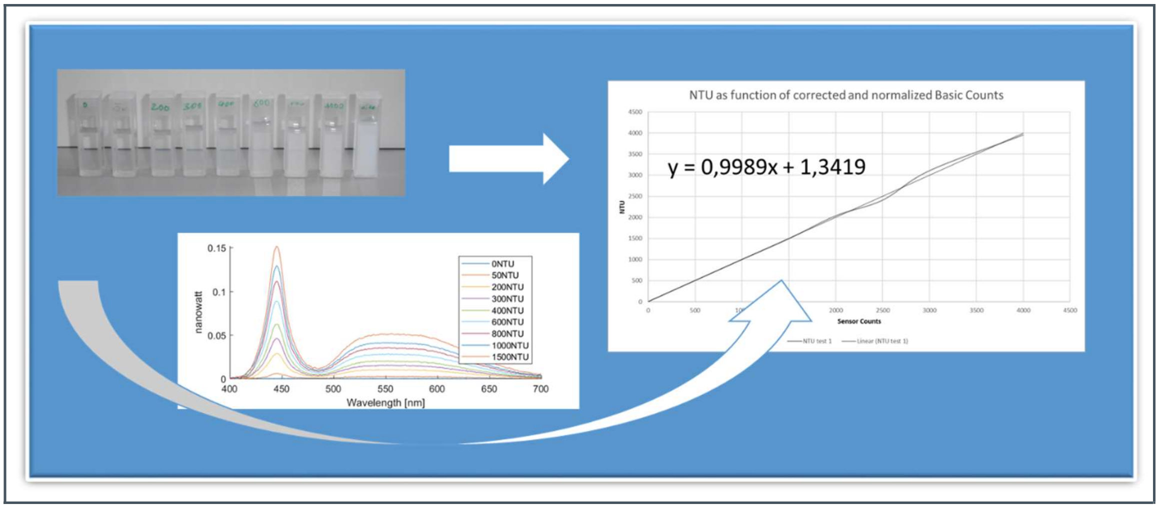
AS7341 Application

v0-01 • 2021-Feb-24

Content Guide

1 Introduction

Figure 1:
From 'Basic Count' to NTU



The following documentation refers to the correction and mapping of sensor data to the 'Turbidity measurement' application, where Sensor results can be used directly to measure NTU (see Figure 1).

The test results were measured with the AS7xxx AquaSensor from **ams** (see [10]) or with the miniLiquid test system [11] of **ams** in combination with AS7341 or AS73261.

Liquids with defined turbidity were available as test objects, which were measured with the sensor and an LED as illumination. The following chapters describe the correction of the sensor data and the generation of the turbidity function based on the obtained sensor data.

The application of the spectral sensor is useful here, since the color and spectrum of the liquid should be measured in parallel with the turbidity. The color measurement is described in [9;10].

The following steps are assumed here as 'already done':

1. Stable setup with sensor and illumination, free from disturbances and drifts [see 10].
2. Installed test software with protocol function [see 10].
3. Optimal parameter setup - integration time, gain level, LED_current [see 10].
4. Installed MS Excel to process the data from the sensor.
5. Measured samples with defined NTU by a lot of sensors - data are logged in the log file as setup independent quantity 'Basic_Counts', which contains corrected and normalized sensor data in the form:

   Basic_Counts normalized per sensor and targets = $\text{Raw\_Counts} / [\text{Gain} = 1];[\text{TINT}=1\text{ms}])$.

2 From Basic_Counts Up to NTU

This chapter describes corrections and how to implement the white-black calibration process (can also be named as 1-0 Normalization or black-white balance) after closing the measurement protocol to get the sensor corrected results and the sensor's uniformity.

These corrected values are the basis to match the sensor results into the application (here for turbidity NTU). This process is named curve fitting and includes the creation and optimization of an equation, which emulates the measured sensor results after the black/white correction as function 'Sensor Counts and NTU'. Then turbidity will be measured from the best-fitted equation.

In this chapter, using an example, the whole procedure about white-black calibration and Curve Fitting will be explained, based on three sensor samples – which match the sensor results from 0 NTU (like pure Water) to 4000 NTU turbidity (like Milk).

2.1 Basic_Counts

After taking the measurement, put all the values from all the sensors into one excel file as shown below.

Figure 2:
#Sensor_1[1]

NTU	X	Y	Z	NIR	DK	CL	TINT [ms]	Gain	LED_current [mA]
0	12798	22824	4899	636	1	60049	607.6	1x	12.5 mA
100	13121	22492	4650	682	1	59477	196	3.7x	12.5 mA
200	13783	22431	4501	719	1	59386	232.4	3.7x	12.5 mA
300	14369	22588	4376	763	1	59866	277.2	3.7x	12.5 mA
500	15193	22243	4180	827	1	59681	364	3.7x	12.5 mA
800	16967	22154	4130	932	1	59947	548.8	3.7x	12.5 mA
1000	17890	21994	4152	992	1	59895	700	3.7x	12.5 mA
1500	19428	21941	4405	1083	1	59768	249.2	16x	12.5 mA
2000	20695	22258	4868	1118	1	59984	347.2	16x	12.5 mA
2500	21395	22513	5319	1104	1	59785	445.2	16x	12.5 mA
3000	21624	22726	5622	1080	1	59819	515.2	16x	12.5 mA
4000	22299	23556	6357	952	1	59361	184.8	64x	12.5 mA

[1] Sensor deviations in series ca. 1% of Lambda; LED deviations > 10nm in peaks (spectral wavelength) and up to 40% brightness

The following graph shows the starting point, which is 0 NTU (White Balance), and ending point 4000 NTU (Black Balance), with the identical coordinates for all sensors.

Figure 3:
#Sensor_2

NTU	X	Y	Z	NIR	DK	CL	TINT [ms]	Gain	LED_current [mA]
0	11108	23304	5513	764	1	59924	624.4	1x	12.5 mA
100	11960	23262	5282	814	1	59873	204.4	3.7x	12.5 mA
200	12556	23123	5056	859	1	59891	240.8	3.7x	12.5 mA
300	13658	23306	4982	852	1	59842	285.6	3.7x	12.5 mA
500	14420	22898	4685	915	1	59530	364	3.7x	12.5 mA
800	16637	23077	4602	1021	1	59782	551.6	3.7x	12.5 mA
1000	17604	23026	4579	1131	1	59925	694.4	3.7x	12.5 mA
1500	19251	22924	4726	1182	1	59459	238	16x	12.5 mA
2000	20597	23260	5130	1229	1	59846	333.2	16x	12.5 mA
2500	21484	23646	5603	1192	1	59878	420	16x	12.5 mA
3000	21862	23907	5901	1179	1	59979	492.8	16x	12.5 mA
4000	22510	24686	6629	967	0	59568	179.2	64x	12.5 mA

Figure 4:
#Sensor_3

NTU	X	Y	Z	NIR	DK	CL	TINT [ms]	Gain	LED_current [mA]
0	19031	24357	6573	663	1	59850	635.6	1x	12.5 mA
100	18810	24290	6040	700	1	59612	204.4	3.7x	12.5 mA
200	19194	24381	5830	720	1	59801	243.6	3.7x	12.5 mA
300	19305	24249	5436	757	1	59743	285.6	3.7x	12.5 mA
500	19758	24051	5044	840	1	60020	375.2	3.7x	12.5 mA
800	20411	23482	4733	915	1	59801	560	3.7x	12.5 mA
1000	20541	23044	4577	969	1	59828	708.4	3.7x	12.5 mA
1500	21323	22654	4732	1079	1	59655	252	16x	12.5 mA
2000	21756	22487	5097	1103	1	59584	350	16x	12.5 mA
2500	22202	22740	5598	1099	1	59792	456.4	16x	12.5 mA
3000	22275	22899	5863	1081	1	59882	518	16x	12.5 mA
4000	22605	23517	6568	936	0	59111	182	64x	12.5 mA

The Basic_Counts are calculated from Raw_Counts (results from the AD-Converters on Chip) with the following formula:

Equation 1:

$$Basic\_Counts = \frac{Raw\_Counts}{Gain * TINT * LED\_current}$$

Basic_Counts are not dependent on the parameter setup and should be used as sensor results in all the following calculations.

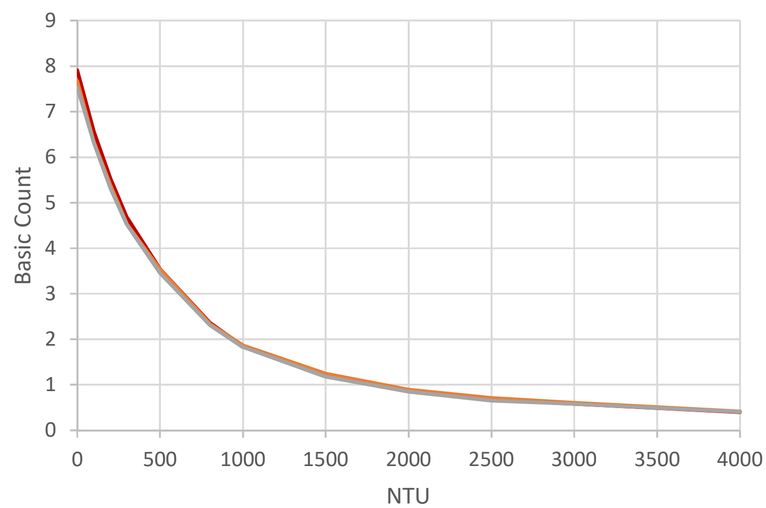
> **NOTE:** Basic_Counts should be corrected if the sensor system has any drift over temperature, time, or by using different gains (see chapter 2.2).

In the table below are Basic_Counts from all three sensors, based on a clear channel for this example. The channel selection must be application-specific and dependent on the used lamination and application, to get maximum counts and high accuracy. An optimum depends on the used luminary, target, and planed accuracy. Select the channel by a maximum of sensor filter luminary overlapping, number of affected filters and slopes, and interferences. The target is to get a maximum number of counts in case of minimal changing of the sensor/luminary/target overlapping.

Figure 5:
Table Measured NTU and Counts of All Sensors

NTU	Sensor_1	Sensor_2	Sensor_3
0	7.90638578	7.677642537	7.53304
100	6.561169333	6.333421484	6.305813
200	5.525050007	5.377660052	5.30786
300	4.66955267	4.530395942	4.522901
500	3.545054945	3.536085536	3.458768
800	2.361791821	2.343335359	2.308919
1000	1.85003861	1.86589239	1.826056
1500	1.199197432	1.249138655	1.183631
2000	0.863824885	0.89804922	0.8512
2500	0.671439802	0.712833333	0.655039
3000	0.580541537	0.608553166	0.578012
4000	0.401521916	0.415513393	0.405982

Figure 6 :
Diagram Measured NTU and Counts for Sensor 1 … Sensor 3 (3 Sensors shown here nearly about each other)



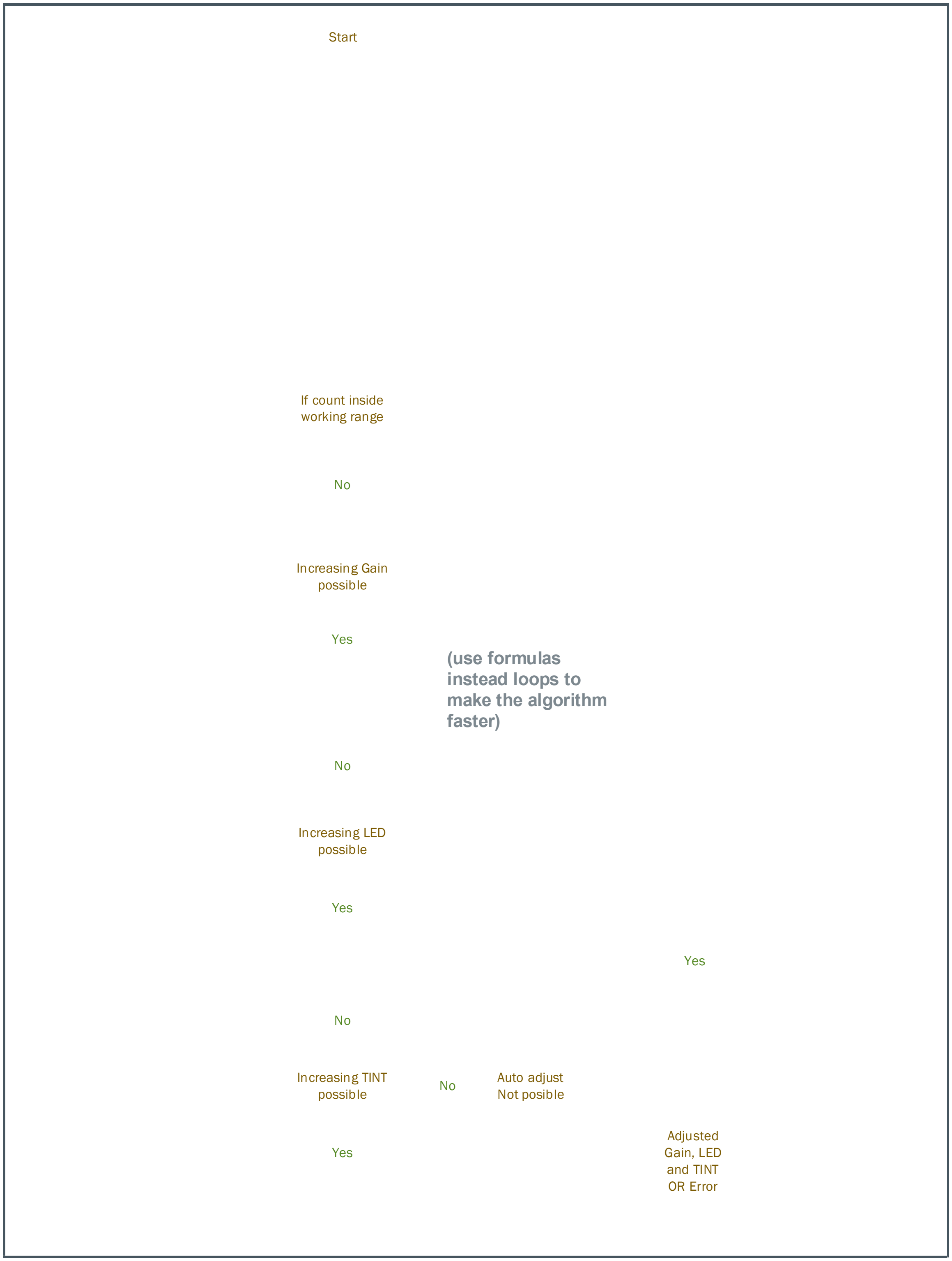
2.2 Dynamic Adjustment and Corrections of Basic_Counts

For applications with a high dynamic range, an automatic adjustment of the conversion parameters is recommended - to ensure that the meters always operate in the optimum working range. The parameters are gain, integration time, and LED driver current. Each of these parameters has its factors, which cause different amplifications but also affect the system results. An increase in LED_currents means a higher temperature of the LED and neighbored elements. Higher gain results in higher noise, but increasing integration time means higher measurement time. Therefore, increasing parameters gain or integration time must be selected application specific by its effects and disadvantages. Besides, since the parameter setting and result are not always exactly linear or vary with temperature, a signal correction must be done in general when using dynamic adjustment.

Automatic adjustment needs a setup to start the process and a target, which represents the workings range. This range will be specified by a hysteresis to prevent an oscillation system.

The dynamic amplification works starting with a setup, its measurement, and the comparison of the results with the optimal working range. Once this has been achieved with the setup, the dynamic process is completed successfully. Otherwise, the parameters are incremented until the optimum working range before saturation is reached. It is recommended to use the parameters in the order of their gain levels to quickly get the sensor out of DARK. Then the working range is adjusted FINE. Since gain works with factor 4, LED_current works with factor 2, and the integration time contains very small steps, this order of the parameters is also recommended. Figure 7 shows the design flow of an automatic conversion which the customer must program in the firmware.

Figure 7:
Design Flow of Automatic Conversion[2]



The result of the automatic adjustment is an optimized setup for the selected parameters to get counts in the specified dynamic range. The next step is the correction of the measured counts based on the used parameters before they will be used in the calibration process. The necessary correction factors must be determined in advance in laboratory tests. Here it is recommended to carry out these tests on many sensors, on as many production batches as possible, and then to work with mean values. A better more accurate method is the device correction, where all parameters for the correction are measured directly on the device. This guarantees the highest accuracy. All tests must be specified and application-specific by their requirements. Therefore, in the following, only two short examples are shown for temperature and gain correction. Other tests are similar.

Making temperature test and correction: Use a stable light source in a climate chamber with the sensor. Set gain and integration time (min 179,2ms) to get counts >5k. Start with a min temperature and log temperature and counts. Measure over the full temperature range up to max temperature. Use these results of counts depending on temperature changing as a lookup table correction factor/°C (e.g. 0°C – 1.0; 10°C – 1.0206; 20°C – 1.0412; ….), to correct temperature drifts with the formula

```
counts_new (of temp e.g. 20°) = counts_new * correction_factor (of 20°C)
```

Making GainError test and correction: Use a stable light source, an integration time (max as possible to get evaluable counts and gain = 1. Start the measurements and increase gain systematically from 1 to 3.7 to 16 to 64. The Gain_error is the factor for each gain change, which is the result of the formula:

```
Gain_factor(gain_new) = (gain_new / gain_old) / (counts_new / counts_old)
```

Use this formula to correct the counts based on the calculated gain_factor.

```
counts_new = counts_new * gain_factor(gain_new)
```

The following figures show a simple example for the correction of two alternative measured counts.

Figure 8 :
Example of Temperature CSV and Gain Correction List

degree	0	1	7	8	9	10	…	81	82	83	84	85
factor	1,000	1,002	1,014	1,016	1,019	1,021	…	1,167	1,169	1,171	1,173	1,175

Gain	Gain Factor	
1	1	The red marked numbers are used in the example in the next figure
3,7	0,9973456	
16	0,9788965	The gain factors were calculated from test results by varying the gains under
64	1,0559696	fixed conditions and comparison with the increasing of the counts

[2] All parameters are examples and were used in tests to get useful results.

Figure 9 :
Example Count Correction as Excel Sheet

	A	B	C	D
1		**Value 1**	**Value 2**	
2	input RAW values - counts	13009	42224	Value 1/2 represents alternative measurements/counts
3	input used Gain	1	3,7	of a constant light source, by using different gains
4	input used TINT in ms	182	182	identical TINT
5	input temp in°C	7	84	but under different temperatures
8				
9	Gain	Gain Faktor		This table is the result of a gain test
10	1	1		
11	3,7	0,9973456		
12	16	0,9788965		
13	64	1,0559696		
14				
15	**Results**	**Value 1**	**Value 2**	
16	**RAW Counts**	13009	42224	=C2
17	**Basic Counts**	71,5	62,7	=C2/($C$3*$C$4)
18	**temp_corrected**	72,4	73,4	=VLOOKUP($C$5;'csv temp'!$A$1:$G$102;2;TRUE)*C17
19	gain_corrected	72,4	73,2	=C18*VLOOKUP($C$3;$A$10:$B$13;2;TRUE)
20	Diff			
21	1,2%	72,4	73,2	the originally differently measured counts are almost completely identical after correction and calculation as basic counts

2.3 Black-White-Balance

Normalization or balancing must be done to eliminate technology and manufacturing deviations in sensing results and to get sensor uniformity. For this example, the table of measured sensor values shows different results of sensor counts between 0 NTU and 4000 NTU. Sensor and LED variations, temperature drifts, and ageing can be the reason for the differences in the sensor results.

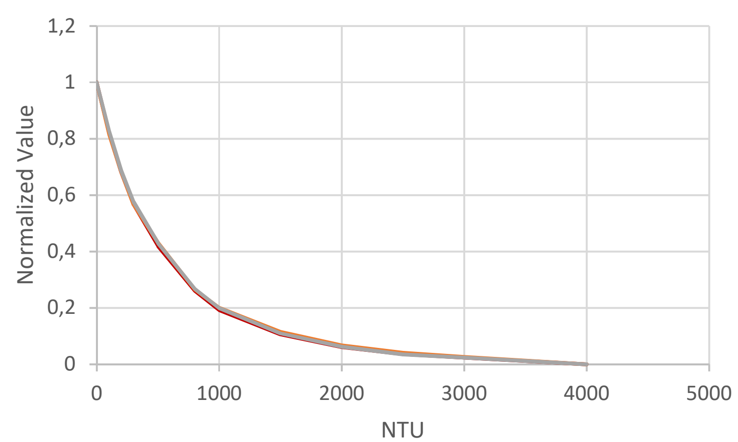
For this example, the Black-White-Balance method was selected to match the sensor results into a defined dynamic range. This range will be the basis for the matching process, which means to find a correlation between sensor results COUNTS and application specific units for turbidity NTU.

After normalization, the table will be as shown below.

Figure 10:
Sensor Results After Black-White-Balance

NTU	Sensor_1	Sensor_2	Sensor_3
0	1	1	1
100	0.820754051	0.814899869	0.827807
200	0.682694341	0.683290886	0.687784
300	0.568701955	0.566622056	0.577646
500	0.418866096	0.429704854	0.428337
800	0.261199929	0.265462363	0.267002
1000	0.193010389	0.19971815	0.199251
1500	0.106287806	0.114790752	0.109112
2000	0.061600447	0.066445503	0.062469
2500	0.035965727	0.040941153	0.034945
3000	0.023853813	0.026581705	0.024138
4000	0	0	0

Figure 11:
Diagram Sensor Results After Black-White-Balance (3 Sensors shown here about each other)



The graph shows the starting point, which is zero NTU (White Balance), and the ending point with 4000 NTU (Black Balance), identical for all sensors but a little different between 0 and 4000 NTU.

> **NOTE:** These differences will affect the accuracy of the sensor system after matching, and should be minimized as much as possible.

The device-specific Black-White-Balance should correct the sensor variations based on the mounted sensor system. Therefore, it is recommended to make the Black-White-Balance during the series end test of the sensor modules. In this case, the measured values for Black and White must be stored on the sensor to be read out by the Sensor's Firmware - if these values are necessary for the calculation of turbidity (see Chapter 2.4).

The counts and/or calculated values for the sensors are the basis for the matching and correlation counts into NTU.

2.4 Curve Fitting

Curve fitting will find out a formula as correlation counts and NTU, based on the normalized values of the Black-White-Balance. MCU needs this formula with the device-specific parameters to calculate NTU values according to sensor-normalized value. The curve fitting process gets the best-fit line or curve for the series of data points (see Figure 11). This curve fitting will produce an equation that can be used to find the points anywhere along the curve.

The curve fitting will be affected by the formula type and its parameters. For the used example, the following formula was fitted as an optimal solution.

Equation 2:

$$y = an * x^n + \ldots. + a2 * x^2 + a1 * x + a0$$

y = calculated NTU
x = counts
a3…a0 = Spec. Sensor parameter from Golden Device, type, lot, or device specific

This formula describes the principal correlation between counts and NTU. The parameters determine the accuracy of the correlation. The better the formula (especially parameters) maps the correlation, the better the result and sensor accuracy will be. Therefore, using a device-specific sensor formula and parameters considers the device-specific deviations and promises the highest accuracy. On the other side, it needs the highest effort in matching and device calibration. A type or lot fitting cannot consider the exemplary deviations and deliver an averaged accuracy by a lower effort.

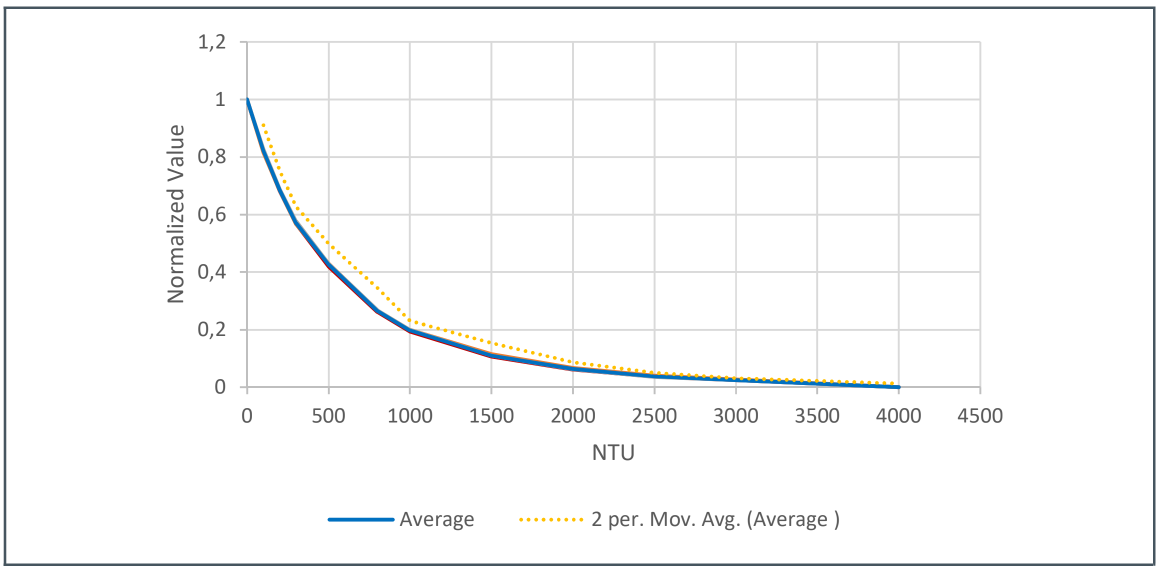
2.4.1 Curve Fitting for Golden Device

In the Golden device calibration, the balanced curves of a Golden Device (in the example, the averaged values from all sensors were used) are used to realize a correlation as a formula that can be used in the sensor's firmware.

Figure 12:
Averaging of Sensor Results

NTU	Sensor_1	Sensor_2	Sensor_3
0	1	1	1
100	0.820754051	0.814899869	0.827807
200	0.682694341	0.683290886	0.687784
300	0.568701955	0.566622056	0.577646
500	0.418866096	0.429704854	0.428337
800	0.261199929	0.265462363	0.267002
1000	0.193010389	0.19971815	0.199251
1500	0.106287806	0.114790752	0.109112
2000	0.061600447	0.066445503	0.062469
2500	0.035965727	0.040941153	0.034945
3000	0.023853813	0.026581705	0.024138
4000	0	0	0

Figure 13 :
Averaging of Sensor Results – Diagram (3 sensors shown here about each other with average and fitted curve)



In the process of curve fitting:

- The inputs are the counts on the x-axis.
- The results will be the matched NTU on the y-axis.

MATLAB was used as a CAD tool for the curve fitting process (like the TRENDLINE function of MS EXCEL in diagrams, see the yellow broken line in Figure 13).

The first tests show a higher accuracy in the case of splitting the curve into three partitions with alternative a0…a3 parameters for the identical function of x-axis [0:500], [500-1500], [1500-4000]. These partitions result in three alternative parameter sets for a0…a3 and must be realized in the sensor's firmware as IF-THEN commands.

The tests with the MATLAB script result in the following for the three sensor diagrams. From 0 NTU to 4000 NTU, there are so many points, a cubic equation is suitable for these. The system will be unstable if the order of the equation will be increased.

Therefore, the equation for our example here is:

Equation 3:

$$y = a3 * x^3 + a2 * x^2 + a1 * x + a0$$

Below is the MATLAB script where we used NTU and Normalization values from the example:

```
% Input data

NTU = [0 100 200 300 500 800 1000 1500 2000 2500 3000 4000];

normalized =[1  0.821153754 0.684589804 0.570990128 0.425636141 0.264554678 0.197326558 0.110063581
0.063504876 0.037284066 0.024857677 0];

%% 0 to 500NTU

x1 = normalized(1:5);

y1 = NTU(1:5);

Output(1,1:4) = polyfit(x1,y1,3);

%% 500 to 1500NTU

x2 = normalized(5:9);

y2 = NTU(5:9);

Output(2,1:4) = polyfit(x2,y2,3);

%% 1500 to 4000NTU

x3 = normalized(9:12);

y3 = NTU(9:12);

Output(3,1:4) = polyfit(x3,y3,3);
```

After executing the script, three equations would have been produced, as shown below.

Figure 14:
MatLab Produced Parameters

NTU	a3	a2	a1	a0
0 - 500	-1488.89	4097.258	-4318.98	1710.256
500 - 1500	-56896.6	53942.54	-18440.5	2963.21
1500 - 4000	8628314	-536391	-32227.1	4000

The following formulas can now be used to describe the correlation between x-counts and y-NTU.

Figure 15:
Results of Sensor Mapping and Calibration

```
IF (x >= 0.425636) THEN y = -1488.89*x^3 + 4097.258*x^2 -4318.98*x + 1710.256

IF (x <= 0.425636) AND (x >=0.110064) THEN
y =-56896.6*x^3 + 53942.54*x^2 -18440.5*x + 2963.21

IF (x < 0.110064) THEN y = 8628314*x^3 -536391*x^2 - 32227.1*x + 4000
```

In the sensor system and firmware (e.g. AS7261 firmware on flash), formulas and parameters can be used directly.

The next step is the verification of the matching for all sensors by using the formulas of the three partitions.

Figure 16:
Golden Device

Normalized Values	NTU	NTU Check	NTU Diff
1	0	-0.35911	0.35911
0.8182	100	103.8473	3.847333
0.681268	200	198.7379	1.26206
0.567806	300	306.3227	6.322699
0.421704	500	505.9002	5.900208
0.260834	800	836.0522	36.05216
0.194446	1000	998.7565	1.243497
0.10871	1500	1522.93	22.9303
0.062691	2000	2005.14	5.140247

Normalized Values	NTU	NTU Check	NTU Diff
0.036702	2500	2521.242	21.24199
0.024433	3000	3018.222	18.22151
0	4000	4000	0

Figure 17:
#Sensor_1

Normalized Values	NTU	NTU Check	NTU Diff
1	0	-0.35911	0.35911
0.820754	100	102.3062	2.30617
0.682694	200	197.5854	2.414579
0.568702	300	305.3333	5.333262
0.418866	500	510.6223	10.62232
0.2612	800	835.1427	35.14269
0.19301	1000	1004.422	4.421568
0.106288	1500	1544.284	44.28363
0.0616	2000	2018.657	18.65721
0.035966	2500	2548.505	48.50461
0.023854	3000	3043.164	43.16442
0	4000	4000	0

Figure 18:
#Sensor_2

Normalized Values	NTU	NTU Check	NTU Diff
1	0	-0.35911	0.35911
0.8149	100	105.8467	5.846743
0.683291	200	197.1047	2.895273
0.566622	300	307.6349	7.634897
0.429705	500	492.7785	7.221452
0.265462	800	824.6109	24.61085
0.199718	1000	978.6772	21.32281
0.114791	1500	1471.144	28.85574
0.066446	2000	1959.386	40.61417
0.040941	2500	2373.615	126.3847
0.026582	3000	2926.402	73.59757
0	4000	4000	0

Figure 19:
#Sensor_3

Normalized Values	NTU	NTU Check	NTU Diff
1	0	0.35911	0.35911
0.827807	100	98.07948	1.920522
0.687784	200	193.5078	6.49219
0.577646	300	295.5878	4.412185
0.428337	500	495.0012	4.998782
0.267002	800	820.8333	20.83332
0.199251	1000	980.4105	19.58945
0.109112	1500	1519.425	19.42547
0.062469	2000	2007.887	7.886994
0.034945	2500	2586.996	86.9955
0.024138	3000	3030.947	30.947
0	4000	4000	0

"NTU Diff" shows, in all tables, the difference between SHOULD and ACTUAL NTU, by using the fitted curve to the Golden Device (here average of sensor 1…3).

2.4.2 Device Calibration

Device calibration considers a correlation function for each Black-White-Balanced curve. That means, more measurements over the full dynamic range of turbidity must be done during the module end test. The results of these measurements are the calculated curve fitting with parameters from the test system, which must be stored on the sensor device. Such a process needs much more effort but guarantees the highest accuracy. Below is the MATLAB script.

```
% Input data

NTU = [0 100 200 300 500 800 1000 1500 2000 2500 3000 4000];

normalized_Sensor_1 =[1 0.820754051 0.682694341 0.568701955 0.418866096 0.261199929 0.193010389
0.106287806 0.061600447 0.035965727 0.023853813 0];

normalized_Sensor_2 =[1 0.814899869 0.683290886 0.566622056 0.429704854 0.265462363 0.19971815
0.114790752 0.066445503 0.040941153 0.026581705 0];

normalized_Sensor_3 =[1 0.827807341 0.687784185 0.577646372 0.428337472 0.267001743 0.199251134
0.109112184 0.062468678 0.034945319 0.024137513 0];

%% 0.2 to 500NTU

x1_Sensor_1 = normalized_Sensor_1(1:5);

y1_Sensor_1 = NTU(1:5);
```

```
Output_Sensor_1(1,1:4) = polyfit(x1_Sensor_1,y1_Sensor_1,3);

x1_Sensor_2 = normalized_Sensor_2(1:5);

y1_Sensor_2 = NTU(1:5);

Output_Sensor_2(1,1:4) = polyfit(x1_Sensor_2,y1_Sensor_2,3);

x1_Sensor_3 = normalized_Sensor_3(1:5);

y1_Sensor_3 = NTU(1:5);

Output_Sensor_3(1,1:4) = polyfit(x1_Sensor_3,y1_Sensor_3,3);

%% 500 to 1500NTU

x2_Sensor_1 = normalized_Sensor_1(5:9);

y2_Sensor_1 = NTU(5:9);

Output_Sensor_1(2,1:4) = polyfit(x2_Sensor_1,y2_Sensor_1,3);

x2_Sensor_2 = normalized_Sensor_2(5:9);

y2_Sensor_2 = NTU(5:9);

Output_Sensor_2(2,1:4) = polyfit(x2_Sensor_2,y2_Sensor_2,3);

x2_Sensor_3 = normalized_Sensor_3(5:9);

y2_Sensor_3 = NTU(5:9);

Output_Sensor_3(2,1:4) = polyfit(x2_Sensor_3,y2_Sensor_3,3);

%% 1500 to 4000NTU

x3_Sensor_1 = normalized_Sensor_1(9:12);

y3_Sensor_1 = NTU(9:12);

Output_Sensor_1(3,1:4) = polyfit(x3_Sensor_1,y3_Sensor_1,3);

x3_Sensor_2 = normalized_Sensor_2(9:12);

y3_Sensor_2 = NTU(9:12);

Output_Sensor_2(3,1:4) = polyfit(x3_Sensor_2,y3_Sensor_2,3);

x3_Sensor_3 = normalized_Sensor_3(9:12);

y3_Sensor_3 = NTU(9:12);

Output_Sensor_3(3,1:4) = polyfit(x3_Sensor_3,y3_Sensor_3,3);
```

After executing the script, parameters for three equations would have been produced, as shown below.

Figure 20:
Parameter for #Sensor_1

NTU	a3	a2	a1	a0
0 - 500	-1323.24	3687.404	-3982.38	1617.91
500 - 1500	-63310.7	58105.49	-19119	2965.753
1500 - 4000	9076520	-525149	-34559.8	4000

Figure 21:
Verification for #Sensor_1

Normalized Values	NTU	NTU Check	NTU Diff
1	0	-0.30293	0.302933
0.820754	100	101.7227	1.722724
0.682694	200	196.7222	3.277821
0.568702	300	302.3258	2.325775
0.418866	500	499.5323	0.467745
0.2612	800	807.9283	7.928325
0.19301	1000	984.9833	15.01666
0.106288	1500	1514.045	14.04499
0.0616	2000	2000	2.59E-07
0.035966	2500	2500	9.04E-06
0.023854	3000	3000	1.45E-05
0	4000	4000	0

Figure 22:
Parameter for #Sensor_2

NTU	a3	a2	a1	a0
0 - 500	-1754.51	4749.629	-4833.98	1838.272
500 - 1500	-51711.7	50813.63	-18081.5	2989.899
1500 - 4000	4715348	-250013	-34305.9	4000

Figure 23:
Verification for #Sensor_2

Normalized Values	NTU	NTU Check	NTU Diff
1	0	-0.58963	0.589628
0.820754	100	103.6669	3.666918
0.682694	200	193.0724	6.927648
0.568702	300	304.9704	4.970436
0.418866	500	498.8799	1.120078
0.2612	800	803.4024	3.402401
0.19301	1000	993.5627	6.437257
0.106288	1500	1505.655	5.654902
0.0616	2000	2000	2.1E-06
0.035966	2500	2500	3.71E-07
0.023854	3000	3000	9.58E-06
0	4000	4000	0

Figure 24:
Parameter for #Sensor_3

NTU	a3	a2	a1	a0
0 - 500	-1429.62	3945.723	-4206.06	1689.792
500 - 1500	-55820.2	52901.74	-18096.1	2931.335
1500 - 4000	13947966	-962403	-26325.6	4000

Figure 25:
Verification for #Sensor_3

Normalized Values	NTU	NTU Check	NTU Diff
1	0	-0.1625	0.162504
0.820754	100	100.8755	0.875452
0.682694	200	198.3115	1.688538
0.568702	300	301.214	1.214038
0.418866	500	499.7616	0.238447
0.2612	800	808.4852	8.485187
0.19301	1000	984.3481	15.65189
0.106288	1500	1514.134	14.13409
0.0616	2000	2000	7.52E-06
0.035966	2500	2500	1.07E-05
0.023854	3000	3000	2.05E-05
0	4000	4000	0

Insertion of these three cubic equations into the firmware, which vary from device to device, will get the actual NTU value from three sensors, according to their normalized values, which are calculated from the Basic count.

2.5 Conclusion

- ☑ To get uniformity for all sensors, Black-White-Balance is necessary.
- ☑ Golden device calibration can be used once for calibration. In this case, an average reading or one standard device has to be selected as a golden device. All other sensor results will be dependent on the Golden device.
- ☑ To get more accuracy, Device calibration is important. In this case, we need to calibrate every device separately.

In case of the results of the sensor system are not in the required range then:

- Check the selection of the LED – sensor combination – via simulation, optimize sensor/luminary overlapping to get a high response for each target (this means the concentrations of substances in any liquid) – compare measurements and simulations to find the reason for non-accuracy.
- Optimize the optical path (limit the output of the LED – FOV FieldOfView) by an optical hole in front of the LED (affects the ratio between transmitted and reflected light).
- In case of lower counts (results of ADC), increase gain/TINT or LED driver currents, use optimized gain, and make sure max. FSR is set (16-bit counter is reserved and max. used).
- Check for drifts that can be corrected (nonlinearities, temperatures).
- Check for other disturbances with tests, where real conditions can be switched on step-by-step (first measurements in the dark without liquid, using known filters to see a clear filter response, then “open” the system step-by-step, by changing the conditions).

3 Additional Documents

The following list includes a selection of available documents with more technical details of the Sensor, AS7341, and its test kits. This list is not fixed and is constantly changing. Ask us for new details.

For further information, please refer to the following documents:

1. ams AG, *AS7341 11-Channel Spectral Sensor Frontend* (DS000504),datasheet.
2. ams AG, *AS7341 Details for Opto-Mechanical Design*, application note.
3. ams AG, *AS7341 Eval Kit Flicker Detection* (AN000605), application note.
4. ams AG, *SMUX configuration* (AN000666), application note.
5. ams AG, *Schematic a0013a_CSS Evalboard AS7341*.
6. ams AG, *AS7341 Eval Kit – Spectral Balance and Calibration* (QG000139), quick start guide.
7. ams AG, *AS7341 Demo for Fast Measurement Using Unicom Board* (AN000660), application note.
8. ams AG, *AS7341 EVK Manual* (UG000400).
9. ams AG, *Calibration* of spectral sensors (AN000633), application note.
10. ams AG, *AS7341 AquaSensor, Measure Color and Spectrum in Liquid.*, quick start guide.
11. ams AG, *MiniLiquid Test System*, quick start guide.

4 Revision Information

Changes from previous version to current revision v0-01	Page
Initial version	all

- Page and figure numbers for the previous version may differ from page and figure numbers in the current revision.
- Correction of typographical errors is not explicitly mentioned.

5 Legal Information

RoHS Compliant & ams Green Statement

RoHS Compliant: The term RoHS compliant means that ams AG products fully comply with current RoHS directives. Our semiconductor products do not contain any chemicals for all 6 substance categories plus additional 4 substance categories (per amendment EU 2015/863), including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, RoHS compliant products are suitable for use in specified lead-free processes.

ams Green (RoHS compliant and no Sb/Br/Cl): ams Green defines that in addition to RoHS compliance, our products are free of Bromine (Br) and Antimony (Sb) based flame retardants (Br or Sb do not exceed 0.1% by weight in homogeneous material) and do not contain Chlorine (Cl not exceed 0.1% by weight in homogeneous material).

Important Information: The information provided in this statement represents ams AG knowledge and belief as of the date that it is provided. ams AG bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. ams AG has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. ams AG and ams AG suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

Headquarters

ams AG

Tobelbader Strasse 30

8141 Premstaetten

Austria, Europe

Tel: +43 (0) 3136 500 0

Please visit our website at www.ams.com

Buy our products or get free samples online at www.ams.com/Products

Technical Support is available at www.ams.com/Technical-Support

Provide feedback about this document at www.ams.com/Document-Feedback

For sales offices, distributors and representatives go to www.ams.com/Contact

For further information and requests, e-mail us at ams_sales@ams.com