Report on mission outputs, describing the raw mapping output derived by submarine sensors in a common geodetic reference frame

Activity 4.1 - Mission planning, data acquisition and storage
WP4 - Implementation of the Georeferenced Open Access Database

SUSHI DROP project (ID 10046731)

Final Version of 22/02/2022

Deliverable Number D.4.1.2

Project Acronym	SUSHIDROP
Project ID Number	10046731
Project Title	SUstainable fiSHeries wIth DROnes data Processing
Priority Axis	3
Specific objective	3.2
Work Package Number	4
Work Package Title	Implementation of the Georeferenced Open Access Database
Activity Number	4.1
Activity Title	Mission planning, data acquisition and storage
Partner in Charge	LP - UNIBO
Partners involved	
Status	Final
Distribution	Public

Table of contents

Abstract 3
Overview 4
RAW Telemetry data 4
Optical Sensor Raw Data 8
MBES Raw Data 11
Abbreviations 17
Acknowledgement 18

Abstract

WP4 is focused to UUV mission planning, data acquisition and storage. This deliverable describes the synergies between the mission planning environment and the acquisition management. Each sensor equipped on board of Blucy produces in output a heterogeneous data stream that are differently recorded on board and/or transmitted via fibre optic cable to the remote station. These data are then recorded in raw format but sequentially ordered, ready to be subsequently processed at the end of the mission generating maps and three-dimensional models.

Overview

An overview of all UUV RAW data acquired during mission is described in this deliverable. The drone samples different types of data:

- information related to telemetry and the Guidance, Navigation and Control system.
- optical data from the use of high-resolution photogrammetric camera and navigation camera; data from acoustic sensors such as multibeam;
- environmental parameters recorded by on-board probes;

RAW Telemetry data

Telemetry is used both to reconstruct the drone's position and status during the mission, and to geo-reference the acquired data for environmental purposes as local measurements for characterizing the underwater environment. This role is particularly played by the information recorded by SVS and CTD probes. In fact, the speed of sound over the entire water column is crucial to improve the information acquired by acoustic sensors, applying Snell's law.

Sensor	Data	Unit of measurements
Inertial Navigation System	Attitude (°): Heading Roll Pitch	°
	Attitude rates : Heading rate Roll Rate Pitch Rate	°/s
	Latitude	90° S to 90° N
	Longitude	180° E to 180° W
	Linear Acceleration	g

	Speed (North and East)	Knot
Doppler Velocity Log (DVL)	Linear Speed	cm/s
	Linear Acceleration	cm/s^2
On board GPS Attitude & Reference System (GPS AHRS)	Latitude	90° S to 90° N
	Longitude	180° E to 180° W
Altimeter	Distance from Bottom	m
Sound Velocity Sensor - Pressure Conductivity Temperature	Sound Velocity	m/s
	Pressure	dBar
	Conductivity	mS/cm
	Temperature	°C
	Depth	m
Ultra Short Baseline System (USBL)	Relative Distance (AUV- Vessel)	m
	Relative Velocity (AUV - Vessel)	m/s
	Relative Attitude	°
	Latitude AUV	180° E to 180° W
	Longitude AUV	90° S to 90° N
Propulsion Controller	RPM	r/min
Batteries	Cells voltage	volt
	Temperature	celsius
	Residual charge	%
GPS Receiver	Latitude Vessel	90° S to 90° N
	Longitude Vessel	180° E to 180° W
	Speed (North and East)	m/s
	Acceleration	m/s^2

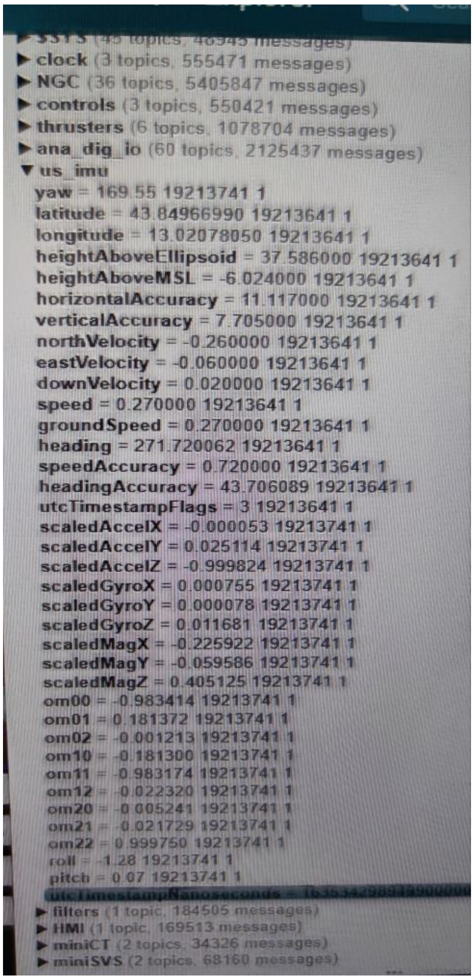
Ultra Short Baseline System (USBL)	Relative Distance (AUV- Vessel)	m
	Relative Velocity (AUV - Vessel)	m/s
	Relative Attitude	°
Sound Velocity Sensor - Pressure Conductivity Temperature	Sound Velocity	m/s
	Pressure	dBar
	Conductivity	mS/cm
	Temperature	°C
	Depth	m

Table 1: Blucy Telemetry Raw Data

An MQTT network protocol is used for telemetry transmission from Blucy to the remote station. MQTT is an efficient and light weight open protocol to deliver publish/subscribe messaging between remote devices, maintained by OASIS Message Queuing Telemetry Transport Technical Committee and ISO/IEC 20922 compilant.

MQTT server is initialized when the drone is turned on and is executed by the main canister CPU. The data is transmitted in real time through the fiber optic cable in ROV mode, or in AUV mode an efficient summary of only the key information for positioning and health of the drone is transmitted through the acoustic channel when the drone is submerged underwater. The remote station computer runs an MQTT client. All information is then displayed in real time and, depending on the mission performed, some of it is recorded and immediately sorted according to the type of sensor used for their acquisition.

The point and telemetry data are saved on a fast SSD disk using a script created ad hoc. Data are recorded within a database formed by a sequential order of text files. Each mission performed produces a new database.



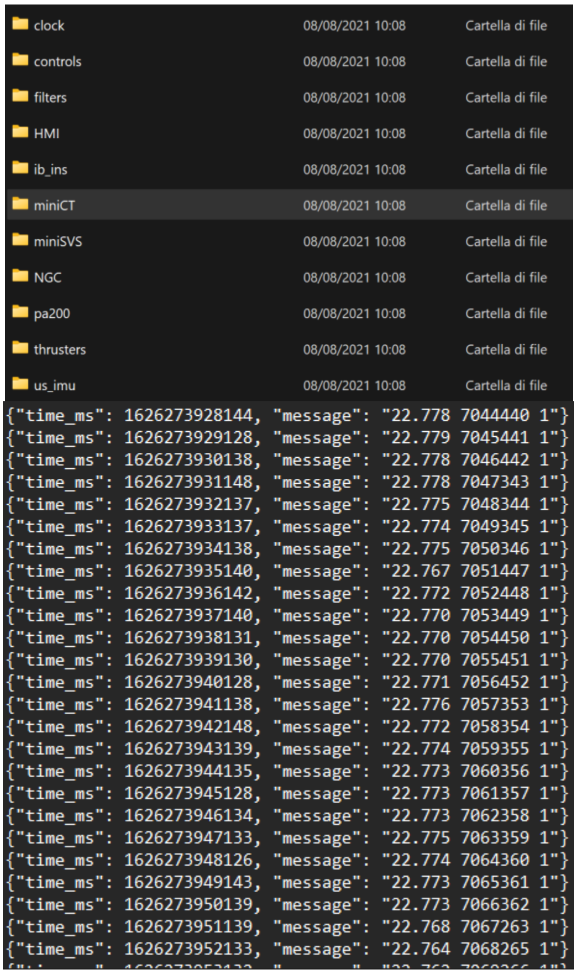


Figure 1 Raw Data Streaming from Blucy to Remote Station via MQTT

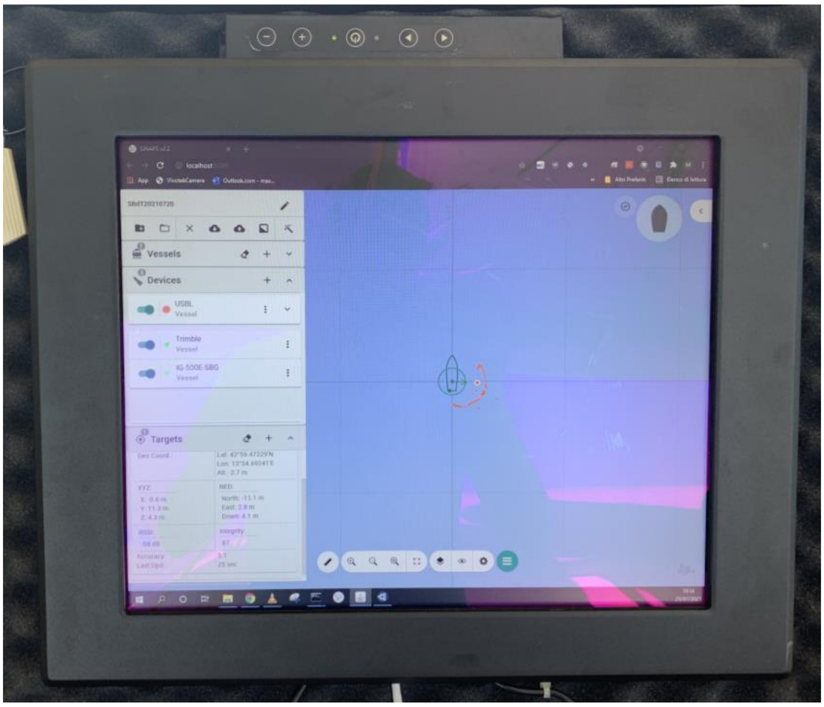


FIGURE 2 RAW POSITION FROM USBL IN SINAPS2

Optical Sensor Raw Data

The optical sensors equipped on Blucy allow to capture single frames and video streams to be transmitted in real time to the remote station through the fiber optic cable when the drone is on mission in ROV mode. The specific orientation of the two optical sensors allows to differentiate the type of acquisition according to a dual approach.

The optical sensor called PilotCAM transmits a video stream with a resolution of 2560x1920 and a constant frequency of 30 fps, compressed with high efficiency h265 coding. The video stream is transmitted over a network protocol and displayed on the remote station to allow manual navigation of the drone. The same video stream is recorded on board the remote station disk and the resulting recording can be used for processing such as automatic recognition of marine features using machine learning algorithms. The PilotCAM is frontally positioned with an adjustable tilt depending on the operational mission.

The optical sensor called BottomCAM consists of a high-resolution full frame camera capable of recording 24-megapixel frame sequences that are transmitted in real time to the remote station through a dedicated fiber optic channel. The image transmission is managed through a subsystem dedicated to the conversion of the USB signal into a network protocol through which the images are rapidly transmitted. The same network interface allows the remote snapshot of frames at the frequency necessary to achieve the photogrammetric coverage of the seabed. In fact, the camera in question is mounted with nadiral attitude with respect to the drone and is specifically designed for the survey of the seabed. By processing the sequence of raw images it is possible to reconstruct three-dimensional models of the seabed according to Structure From Motion procedures.

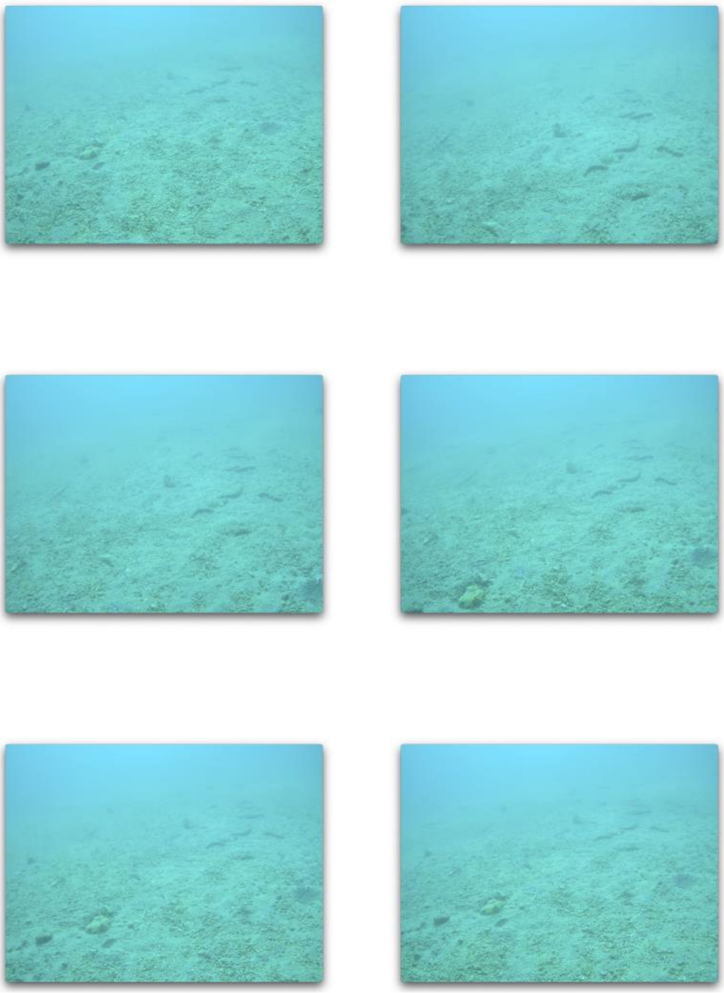


FIGURE 3 PILOTCAM RAW DATA

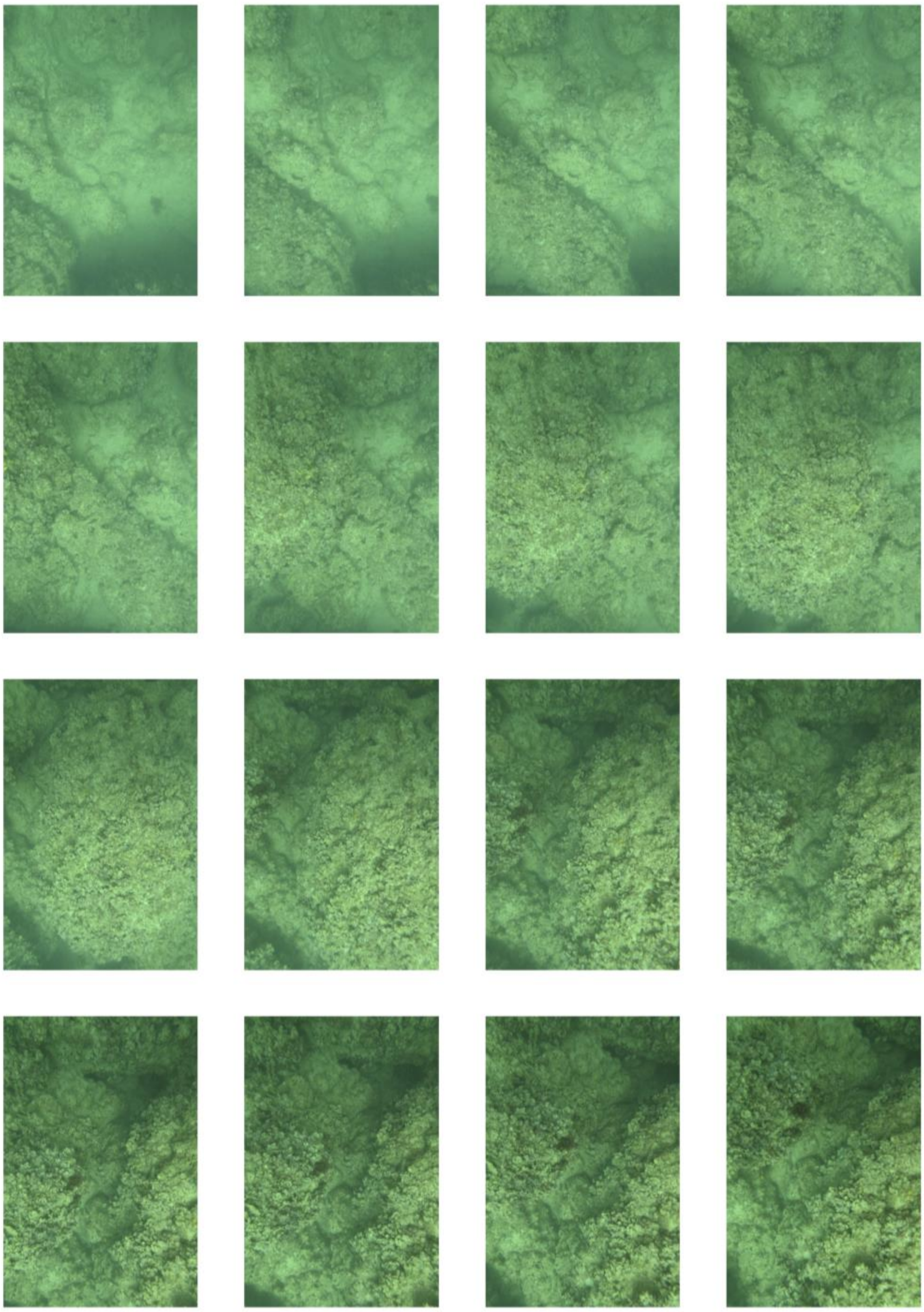


FIGURE 4 SEQUENCE OF FRAMES FROM BOTTOMCAM

MBES Raw Data

The R2 Sonic 2020 multibeam is capable of collecting two types of acoustic data:

- Bathymetry is a process of acquiring the depth of the sea floor in discrete points. The greater the number of points where this phenomenon occurs, the higher the resolution of the final image. In this way it is possible to get information about the morphology of the seabed and what is on it.
- WaterColumn data represent what is present in the water column between the sonar and the seabed. With this type of information it is possible to detect the presence of objects, large fish and schools of fish.

Data are saved on the Remote station in pcap format and are organized in well-defined structures where each type of data is divided into specific substructures, each with its own header. All data that comes from the MBES is grouped in UDP/IP packets and consists of the following types of values:

- u8, u16, u32 = denote unsigned integers of 8, 16, and 32 bits;
- s8, s16, s32 = indicate signed integers of 8, 16 and 32 bits;
- f32 = indicate 32-bit floating point numbers.


```
0000  00 1b 21 ad 79 09 00 50  c2 90 41 e9 08 00 45 00   ··!·y··P ··A···E·
0010  04 b8 a4 d1 00 00 32 11  c9 a8 0a 00 00 56 0a 00   ······2· ·····V··
0020  01 66 ff e1 9c 45 04 a4  00 00 57 43 44 30 00 00   ·f···E·· ··WCD0··
0030  04 9c 00 00 00 00 48 30  00 74 32 30 32 32 00 00   ······H0 ·t2022··
0040  00 00 00 00 00 00 31 30  30 33 37 37 00 00 00 00   ······10 0377····
0050  00 00 00 00 03 ed 08 d2  24 e2 00 00 13 fc 3e 4c   ········ $·····>L
0060  cc cd 44 bd 60 00 48 c3  50 00 43 45 00 00 37 a7   ··D·`·H· P·CE··7·
0070  c5 ac 3c 8e fa 36 3c 0e  fa 36 00 00 00 00 00 00   ··<··6<· ·6······
0080  00 00 00 00 00 00 47 6a  60 00 47 85 97 ba 40 a0   ······Gj `·G···@·
0090  00 00 41 50 00 00 00 00  00 00 00 00 00 00 00 00   ··AP···· ········
00a0  00 00 00 00 00 00 00 00  01 00 41 31 04 1c 00 00   ········ ··A1····
00b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00c0  00 00 00 00 00 00 bf a5  c8 d3 bf a4 7b f4 bf a3   ········ ····{···
00d0  2f 16 bf a1 e2 37 bf a0  95 59 bf 9f 48 7a bf 9d   /····7·· ·Y··Hz··
00e0  fb 9c bf 9c ae bd bf 9b  61 df bf 9a 15 00 bf 98   ········ a·······
00f0  c8 22 bf 97 7b 43 bf 96  2e 65 bf 94 e1 86 bf 93   ·"··{C·· .e······
0100  94 a8 bf 92 47 c9 bf 90  fa ea bf 8f ae 0c bf 8e   ····G··· ········
0110  61 2d bf 8d 14 4f bf 8b  c7 70 bf 8a 7a 92 bf 89   a-···O·· ·p··z···
0120  2d b3 bf 87 e0 d5 bf 86  93 f6 bf 85 47 18 bf 83   -······· ····G···
0130  fa 39 bf 82 ad 5b bf 81  60 7c bf 80 13 9e bf 7d   ·9···[·· `|·····}
0140  8d 7f bf 7a f3 c2 bf 78  5a 04 bf 75 c0 47 bf 73   ···z···x Z··u·G·s
0150  26 8a bf 70 8c cd bf 6d  f3 10 bf 6b 59 53 bf 68   &··p···m ···kYS·h
0160  bf 96 bf 66 25 d9 bf 63  8c 1c bf 60 f2 5f bf 5e   ···f%··c ···`·_·^
0170  58 a2 bf 5b be e5 bf 59  25 28 bf 56 8b 6b bf 53   X··[···Y %(·V·k·S
0180  f1 ae bf 51 57 f1 bf 4e  be 34 bf 4c 24 77 bf 49   ···QW··N ·4·L$w·I
0190  8a ba bf 46 f0 fd bf 44  57 40 bf 41 bd 83 bf 3f   ···F···D W@·A···?
01a0  23 c6 bf 3c 8a 09 bf 39  f0 4c bf 37 56 8f bf 34   #··<···9 ·L·7V··4
01b0  bc d2 bf 32 23 15 bf 2f  89 58 bf 2c ef 9b bf 2a   ···2#··/ ·X·,···*
01c0  55 de bf 27 bc 21 bf 25  22 63 bf 22 88 a6 bf 1f   U··'·!·% "c·"····
01d0  ee e9 bf 1d 55 2c bf 1a  bb 6f bf 18 21 b2 bf 15   ····U,·· ·o··!···
01e0  87 f5 bf 12 ee 38 bf 10  54 7b bf 0d ba be bf 0b   ·····8·· T{······
01f0  21 01 bf 08 87 44 bf 05  ed 87 bf 03 53 ca bf 00   !····D·· ····S···
0200  ba 0d be fc 40 a0 be f7  0d 26 be f1 d9 ac be ec   ····@··· ·&······
0210  a6 32 be e7 72 b8 be e2  3f 3e be dd 0b c4 be d7   ·2··r··· ?>······
```


FIGURE 5 MBES PCAP FILE

The bathymetric data are composed of nine substructures, each with its own header of recognition. The first information is the definition of the organization of all the data that will be represented and the total size of the packet for the type of structure defined. This fundamental information are contained in the lines

- u32 PacketName; // 'BTH0'
- u32 PacketSize; // [bytes] size of this entire packet
- u32 DataStreamID; // reserved for future use

The third line indicates the possibility to find some values that are not used but that are probably present for future updates of data structures. Next we have the section H0: Header, that part of data where all sonar settings for that specific measurement are represented. In this section you have different types of data, signed, unsigned and floating point. Then we have the R0 section where all the data related to the range considered for that specific range considered for that specific measurement. Sections A0 and A2 are sections that may not appear, depending on some settings set on the multibeam. As indicated above, section A0 appears only in case of equi-angle measurements while section A2 appears only in case of equi-distant measurements. Also section I1 is present only in case of enabling the relative function on the MBES control software. The sections G0 and Q0 are always present and are structured exactly like the others. The G1 section is present only when the verbose mode is active. Regardless of the section, at the beginning of each one there are data that allow to understand immediately the name and the size of the substructure that is being examined. In fact, the first two fields SectionName and SectionSize are always present. The data structure of the bathymetry is shown in the following:

6. Bathymetry Data Format

```
// *** BEGIN PACKET: BATHY DATA FORMAT Ø ***

  u32  PacketName;                  // 'BTHØ'
  u32  PacketSize;                  // [bytes] size of this entire packet
  u32  DataStreamID;                // reserved for future use

  // section HØ: header

  u16  HØ_SectionName;              // 'HØ'
  u16  HØ_SectionSize;              // [bytes] size of this entire section
  u8   HØ_ModelNumber[12];          // example "2024", unused chars are nulls
  u8   HØ_SerialNumber[12];         // example "100017", unused chars are nulls
  u32  HØ_TimeSeconds;              // [seconds] ping time relative to ØØØØ hours 1-Jan-197Ø, integer part
  u32  HØ_TimeNanoseconds;          // [nanoseconds] ping time relative to ØØØØ hours 1-Jan-197Ø, fraction part
  u32  HØ_PingNumber;               // pings since power-up or reboot
  f32  HØ_PingPeriod;               // [seconds] time between most recent two pings
  f32  HØ_SoundSpeed;               // [meters per second]
  f32  HØ_Frequency;                // [hertz] sonar center frequency
  f32  HØ_TxPower;                  // [dB re 1 uPa at 1 meter]
  f32  HØ_TxPulseWidth;             // [seconds]
  f32  HØ_TxBeamwidthVert;          // [radians]
  f32  HØ_TxBeamwidthHoriz;         // [radians]
  f32  HØ_TxSteeringVert;           // [radians]
  f32  HØ_TxSteeringHoriz;          // [radians]
  u16  HØ_TxMiscInfo;               // reserved for future use
  s16  HØ_VTX+Offset;               // [hundredths of a dB] transmit voltage offset at time of ping (divide value by 1ØØ to get dB)
  f32  HØ_RxBandwidth;              // [hertz]
  f32  HØ_RxSampleRate;             // [hertz] sample rate of data acquisition and signal processing
  f32  HØ_RxRange;                  // [meters] sonar range setting
  f32  HØ_RxGain;                   // [multiply by two for relative dB]
  f32  HØ_RxSpreading;              // [dB (times log range in meters)]
  f32  HØ_RxAbsorption;             // [dB per kilometer]
  f32  HØ_RxMountTilt;              // [radians]
  u32  HØ_RxMiscInfo;               // reserved for future use
  u16  HØ_reserved;                 // reserved for future use
  u16  HØ_Points;                   // number of bathy points

  // section RØ: 16-bit bathy point ranges

  u16  RØ_SectionName;              // 'RØ'
  u16  RØ_SectionSize;              // [bytes] size of this entire section
  f32  RØ_ScalingFactor;
  u16  RØ_Range[HØ_Points];         // [seconds two-way] = RØ_Range * RØ_ScalingFactor
  u16  RØ_unused[HØ_Points & 1];    // ensure 32-bit section size

  // section AØ: bathy point angles, equally-spaced (present only during "equi-angle" spacing mode)

  u16  AØ_SectionName;              // 'AØ'
  u16  AØ_SectionSize;              // [bytes] size of this entire section
  f32  AØ_AngleFirst;               // [radians] angle of first (port side) bathy point, relative to array centerline, AngleFirst < AngleLast
  f32  AØ_AngleLast;                // [radians] angle of last (starboard side) bathy point
  f32  AØ_MoreInfo[6];              // reserved for future use
```

```
// section A2: 16-bit bathy point angles, arbitrarily-spaced (present only during "equi-distant" spacing mode)

  u16  A2_SectionName;             // 'A2'
  u16  A2_SectionSize;             // [bytes] size of this entire section
  f32  A2_AngleFirst;              // [radians] angle of first (port side) bathy point, relative to array centerline, AngleFirst < AngleLast
  f32  A2_ScalingFactor;
  f32  A2_MoreInfo[6];             // reserved for future use
  u16  A2_AngleStep[H0_Points];    // [radians] angle[n] = A2_AngleFirst + (32-bit sum of A2_AngleStep[0] through A2_AngleStep[n]) * A2_ScalingFactor
  u16  A2_unused[H0_Points & 1];   // ensure 32-bit section size

  // section I1: 16-bit bathy intensity (present only if enabled)

  u16  I1_SectionName;             // 'I1'
  u16  I1_SectionSize;             // [bytes] size of this entire section
  f32  I1_ScalingFactor;
  u16  I1_Intensity[H0_Points];    // [micropascals] intensity[n] = I1_Intensity[n]) * I1_ScalingFactor
  u16  I1_unused[H0_Points & 1];   // ensure 32-bit section size

  // section G0: simple straight-line depth gates

  u16  G0_SectionName;             // 'G0'
  u16  G0_SectionSize;             // [bytes] size of this entire section
  f32  G0_DepthGateMin;            // [seconds two-way]
  f32  G0_DepthGateMax;            // [seconds two-way]
  f32  G0_DepthGateSlope;          // [radians]

  // section G1: 8-bit gate positions, arbitrary paths (present only during "verbose" gate description mode)

  u16  G1_SectionName;             // 'G1'
  u16  G1_SectionSize;             // [bytes] size of this entire section
  f32  G1_ScalingFactor;
  struct
  {
    u8   RangeMin;                 // [seconds two-way] = RangeMin * G1_ScalingFactor
    u8   RangeMax;                 // [seconds two-way] = RangeMax * G1_ScalingFactor
  } G1_Gate[H0_Points];
  u16  G1_unused[H0_Points & 1];   // ensure 32-bit section size

  // section Q0: 4-bit quality flags

  u16  Q0_SectionName;             // 'Q0' quality, 4-bit
  u16  Q0_SectionSize;             // [bytes] size of this entire section
  u32  Q0_Quality[(H0_Points+7)/8]; // 8 groups of 4 flags bits (phase detect, magnitude detect, reserved, reserved), packed left-to-right

// *** END PACKET: BATHY DATA FORMAT 0 ***
```

FIGURE 6 BATHIMETRY RAW DATA FORMAT

The WaterColumn data have 5 sub-structures. As in the case of bathymetry, the first lines define the organization of all data with reference to their name and size. Also here the next section is the one defined by H0:Header where all the sonar-related settings are indicated. Following there is section A1 and then sections M1 and M2, where the return echo values for the water column representation are shown. The WaterColumn data structure is shown below:

8. Water Column (WC) Data Format

```
// *** BEGIN PACKET: WATER COLUMN (WC) DATA FORMAT Ø ***

  // The water column data contains real-time beamformer 16-bit magnitude data
  // (beam amplitude) and optional 16-bit split-array phase data (intra-beam
  // direction). Maximum data rate is about 7Ø megabytes per second (assuming
  // 256 beams, 68.4 kHz sample rate, and phase data enabled). The sample rate
  // (and signal bandwidth) varies with transmit pulse width and range setting.
  // Maximum ping data size is about 32 megabytes (assuming 256 beams of 32768
  // samples, and phase data enabled), but max size may change in the future.
  // The number of beamformed data samples normally extends somewhat further
  // than the user's range setting.
  //
  // When the operator enables water column mode, each sonar ping outputs
  // numerous 'WCDØ' packets containing: one HØ header section, one A1 beam
  // angle section, and many M1 or M2 data sections. The section order may
  // change in the future, so plan for that in your data acquisition.
  //
  // Each M1 or M2 section contains a subset of the ping data. Its header
  // indicates its size position to help you assemble the full ping array.
  //
  // You may wish to detect missing M1 or M2 data sections (perhaps a lost
  // UDP packet), and then fill the gap with zeros or perhaps data from the
  // previous ping (to reduce visual disturbances), and then increment an
  // error counter for network health monitoring purposes.
  //
  // The water column data is basically in polar coordinates, so you may
  // wish to geometrically warp it into the familiar wedge shape for display.
  // Consider using OpenGL or Direct3D texture mapping.

  u32  PacketName;                 // 'WCDØ'
  u32  PacketSize;                 // [bytes] size of this entire packet
  u32  DataStreamID;               // reserved for future use

  // section HØ: header (only one per ping)

  u16  HØ_SectionName;             // 'HØ'
  u16  HØ_SectionSize;             // [bytes] size of this entire section
  u8   HØ_ModelNumber[12];         // example "2024", unused chars are nulls
  u8   HØ_SerialNumber[12];        // example "1ØØØ17", unused chars are nulls
  u32  HØ_TimeSeconds;             // [seconds] ping time relative to ØØØØ hours 1-Jan-197Ø, integer part
  u32  HØ_TimeNanoseconds;         // [nanoseconds] ping time relative to ØØØØ hours 1-Jan-197Ø, fraction part
  u32  HØ_PingNumber;              // pings since power-up or reboot
  f32  HØ_PingPeriod;              // [seconds] time between most recent two pings
  f32  HØ_SoundSpeed;              // [meters per second]
  f32  HØ_Frequency;               // [hertz] sonar center frequency
  f32  HØ_TxPower;                 // [dB re 1 uPa at 1 meter]
  f32  HØ_TxPulseWidth;            // [seconds]
  f32  HØ_TxBeamwidthVert;         // [radians]
  f32  HØ_TxBeamwidthHoriz;        // [radians]
  f32  HØ_TxSteeringVert;          // [radians]
  f32  HØ_TxSteeringHoriz;         // [radians]
  u16  HØ_TxMiscInfo;              // reserved for future use
  s16  HØ_VTX+Offset;              // [hundredths of a dB] transmit voltage offset at time of ping (divide value by 1ØØ to get dB)
  f32  HØ_RxBandwidth;             // [hertz]
  f32  HØ_RxSampleRate;            // [hertz] sample rate of data acquisition and signal processing
  f32  HØ_RxRange;                 // [meters] sonar range setting
  f32  HØ_RxGain;                  // [multiply by two for relative dB]
  f32  HØ_RxSpreading;             // [dB (times log range in meters)]
  f32  HØ_RxAbsorption;            // [dB per kilometer]
  f32  HØ_RxMountTilt;             // [radians]
```

```
u32  H0_RxMiscInfo;               // reserved for future use
u16  H0_reserved;                 // reserved for future use
u16  H0_Beams;                    // number of beams

// section A1: float beam angles, arbitrarily-spaced (only one per ping)

u16  A1_SectionName;              // 'A1'
u16  A1_SectionSize;              // [bytes] size of this entire section
f32  A1_MoreInfo[6];              // reserved for future use
f32  A1_BeamAngle[H0_Beams];      // [radians] angle of beam relative to array centerline, ordered from port to starboard, first angle < last angle

// section M1: 16-bit magnitude data (present only during "magnitude-only" water column data mode, many per ping, you assemble them into complete ping

u16  M1_SectionName;              // 'M1'
u16  M1_SectionSize;              // [bytes] size of this entire section
u32  M1_PingNumber;               // pings since power-up or reboot
f32  M1_ScalingFactor;            // reserved for future use
u32  M1_TotalSamples;             // range samples in entire ping, sample rate is H0_RxSampleRate
u32  M1_FirstSample;              // first sample of this section
u16  M1_Samples;                  // number of samples in this section
u16  M1_TotalBeams;               // beams (always a multiple of 2) (typically columns in your memory buffer)
u16  M1_FirstBeam;                // first beam of this section (always a multiple of 2)
u16  M1_Beams;                    // number of beams in this section (always a multiple of 2)
u32  M1_reserved0;                // reserved for future use
u32  M1_reserved1;                // reserved for future use
struct
{
  u16  magnitude;                 // values 0 to 65535 map non-linearly (due to TVG scaling and possible gain compression) to signal amplitude
} M1_Data[M1_Beams][M1_Samples];  // magnitude data (typical example: 256 beams each containing 36 two-byte structs, 16 kilobytes)

// section M2: 16-bit magnitude and phase data (present only during "magnitude and phase" water column data mode, many per ping, you assemble them int
// complete ping data)

u16  M2_SectionName;              // 'M2'
u16  M2_SectionSize;              // [bytes] size of this entire section
u32  M2_PingNumber;               // pings since power-up or reboot
f32  M2_ScalingFactor;            // reserved for future use
u32  M2_TotalSamples;             // range samples in entire ping, sample rate is H0_RxSampleRate
u32  M2_FirstSample;              // first sample of this section
u16  M2_Samples;                  // number of samples in this section
u16  M2_TotalBeams;               // beams (always a multiple of 2) (typically columns in your memory buffer)
u16  M2_FirstBeam;                // first beam of this section (always a multiple of 2)
u16  M2_Beams;                    // number of beams in this section (always a multiple of 2)
u32  M2_reserved0;                // reserved for future use
u32  M2_reserved1;                // reserved for future use
struct
{
  u16  magnitude;                 // values 0 to 65535 map non-linearly (due to TVG scaling and possible gain compression) to signal amplitude
  s16  phase;                     // values -32768 to +32767 map non-linearly (due to complex transfer function) to target angle within the beamwidth
} M2_Data[M2_Beams][M2_Samples];  // magnitude and phase data (typical example: 256 beams each containing 36 four-byte structs, 36 kilobytes)

/ *** END PACKET: WATER COLUMN (WC) DATA FORMAT 0 ***
```

FIGURE 7 WATERCOLUMN RAW DATA FORMAT

These types of data will be onerous to transmit depending on the functions activated. The following table shows the data rates of the three main types of data generated by the sonar. For WaterColumn data it is also possible to reduce the data rate by increasing the width of the pulse. At certain pulse widths, in fact, the receiver sampling rate is halved, which halves the WaterColumn data rate

Data acquired	Data rate
Bathymetry	800 kb/s
Water Column	560 Mb/s (Mag+Phase mode) 280MB/s (Mag mode)

TABLE 2 MBES DATA RATE

Abbreviations

The following abbreviations are used in this deliverable:

AHRS	Attitude and Heading Reference System
AUV	Autonomous Underwater Vehicle
DEM	Digital Elevation Model
DT	Digital Twin
DVL	Doppler Velocity Log
EKF	Extended Kalman Filter
FDI	Fault Detection and Isolation
FOG	Fiber Optic Gyroscope
FOV	Field of View
GNSS	Global Navigation Satellite System
GSD	Ground Sample Distance
INS	Inertial Navigation System
MBES	Multibeam Echosounder
NGC	Navigation, Guidance and Control
ROV	Remotely Operated Vehicle
RS	Remote Station
SFM	Structure from Motion
USBL	Ultra-Short Baseline
UUV	Unmanned Underwater Vehicle
GIS	Geographic Information System
CoG	Centre of Gravity
NMEA	National Marine Electronic Association
UDP	User Data Protocol

Interreg
Italy - Croatia
SUSHI DROP

EUROPEAN UNION

Acknowledgement

The material in this document is published in "Underwater Drone Architecture for Marine Digital Twin: Lessons Learned from SUSHI DROP Project" by Lambertini, A.; Menghini, M.; Cimini, J.; Odetti, A.; Bruzzone, G.; Bibuli, M.; Mandanici, E.; Vittuari, L.; Castaldi, P.; Caccia, M.; De Marchi, L. in *Sensors* **2022**, *22*, 744. https://doi.org/10.3390/s22030744