Video Surveillance and Monitoring Communication Specification Document 98-2.1

Alan Lipton
Carnegie Mellon University

Yun-Ching Lee
Carnegie Mellon University

Terry Boult
Lehigh University

July 9, 1998

Contents

1 Contact details

For any further information or feedback, please contact:

Dr Alan Lipton
The Robotics Institute
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA, 15213

Phone: (412) 268 1450
Fax: (412) 268 5571
Email: ajl@cs.cmu.edu
Url: http://www.cs.cmu.edu/˜ ajl

2 Overview - VSAM communication notional architecture

The VSAM network represents a new paradigm of sensor cooperation and control and thus requires new communications solutions. The system must be able to support such diverse modalities as full internet video streaming on the one hand, down to symbolic video extracted data transmitted over low bandwidth unreliable radio links on the other. It must support smart sensors and no-so-smart sensors; complex sensor cooperation and control strategies; variable sensor modalities and long-term operation.

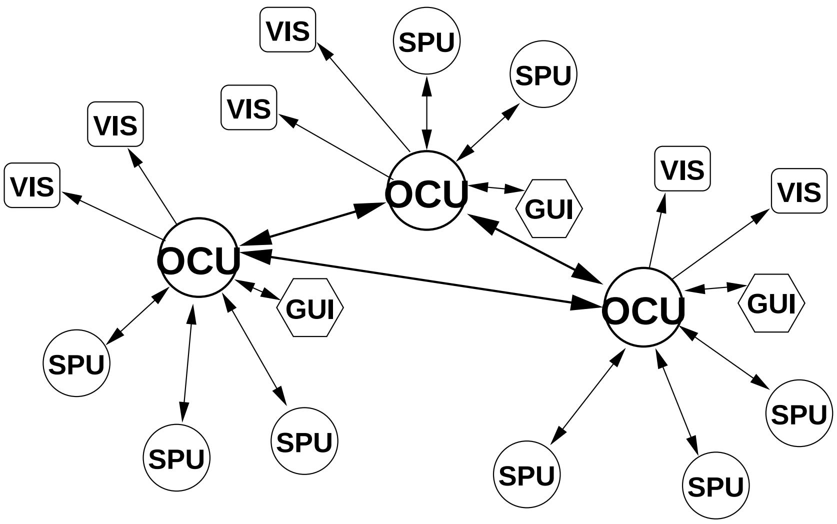

Figure 1: *A nominal architecture for expandable VSAM networks. Operator control units (OCUs) are connected together by high bandwidth data links. Each one has control over several organic sensor processing units (SPUs). Control is provided by a user on a connected GUI. In addition, sensor network data is disseminated through series a of visualisation (VIS) nodes.*

Figure 1 depicts a nominal architecture for a VSAM network. It allows multiple VSAM operator control units (OCUs) to be linked together while continuing to control local sensor processing units (SPUs). Also, data dissemination is not limited to a single user interface, but is accessible to a series of visualisation nodes (VIS). SPUs can be connected to the system by radio links, serial lines, wireless ethernet, cables, or any other medium. Furthermore, arrangement of SPUs can be based on physical proximity to their host OCU, sensor type, communication type or any other criterion.

2.1 Notional architecture

Four types of VSAM node will be supported by this architecture: OCUs, SPUs, GUIs and VIS nodes. Each OCU will support extacly one GUI through which all user related command and control information is passed. However, each OCU can support multiple SPUs and VIS nodes. SPUs are generic sensor processing units which can vary from simple sensors with rudimentary vision processing capabilities to very sophisticated sensing systems capable of making intelligent inferences about activities in their field of regard. VIS nodes are designed to distribute the output of the VSAM network to where it is needed. They will be able to provide symbolic representations of detected activities as well as some simple associated imagery. As far as the protocol is concerned, OCUs, GUIs and SPUs are indistinguishable. However, it is expected that VSAM sensor networks will be used to support the distributed simulation (DIS) community so all VIS nodes will be assumed to be DIS compatible.

Consequently, there will two independent communication protocols and packet structures supported in this architecture: the Carnegie Mellon Packet Architecture (CMUPA) and the DIS packet architecture. The CMUPA is designed to be a low bandwidth, highly flexible architecture in which relevant VSAM information can be compactly packed without, redundant packet overhead. All communication between SPUs, OCUs and GUIs will be CMUPA compatible. Information flow to VIS nodes will be unidirectional, and will originate from OCUs. All of this communication will use the DIS protocol.

In addition to this, there will be (in future implementations) information available to newcomers about the status of the network. The position, IP address and capacity of each of the OCUs will be available so that new SPUs and VIS nodes can automatically make informed decisions about which OCU to connect to. This will allow the VSAM network to automatically configure itself so as to balance the computational load on individual OCUs.

This document describes, in detail, the CMUPA. The relevant coordinate systems are defined, the data which is representable is presented, and the CMUPA packet structure is described along with detailed instructions about data packing. The DIS protocol is described in detail in the DIS specification document, but the VSAM DIS packet structures are presented here as well.

3 VSAM packet architecture (CMUPA)

The notional architecture of the VSAM network requires a very flexible, yet small packet structure. To this end, the Carnegie Mellon Packet Architecture (CMUPA) is proposed. The concept of the packet architecture is a hierarchical decomposition. There are five data sections which could be encoded into a packet: command, sensor, image, target, event. There is a short packet header section which describes which of these five sections are present in the packet. Within each of these sections, it is possible to represent multiple instances of that type of data. So, for example, the target section may contain information about 4 different targets. Each of these information blocks also contains a short header which describes exactly what information is present in the block. One target block may contain position information about a target, whereas a different one (encoded in the same packet) may contain an image template of a different target. At each level, short bitmasks are used to describe the contents of the various blocks within the packets. This way, minimum space is wasted.

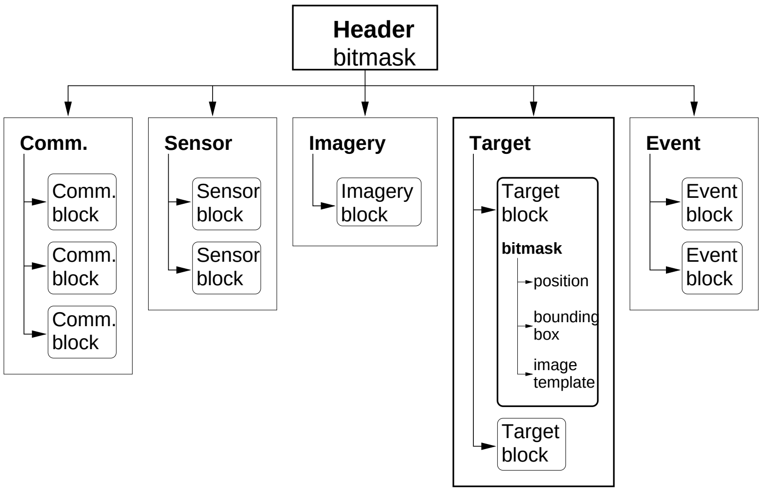

Figure 2: *A bitmask in the header describes which sections are present in the packet. Within each section, multiple data blocks can be present. Within each data block, bitmasks describe which information is present.*

4 Coordinate systems

All quantities which are described in this protocol are represented in their most "natural" coordinates. However, in some cases, there are several possibilities for representation. In these cases, an effort has been made to support all appropriate coordinate systems. In any case, four coordinate systems have been identified as relevant to VSAM applications: image coordinates, pan/tilt spherical coordinates, a Local Vertical Coordinate System (LVCS), and geodetic WGS-84 coordinates (GPS).

Coordinate system	Position dimensions	Rotational dimensions	Time
Image coordinates	(u, v, r)		t
pan/tilt coordinates	r	$(\theta_{\text{pan}}, \phi_{\text{tilt}}, \rho_{\text{twist}})$	t
LVCS	(x, y, z)	(θ, ϕ, ρ)	t
GPS	(Lat, Lon, Alt)	(θ, ϕ, ρ)	t

Table 1: *Nomenclature for coordinate system dimensions*

4.1 Image coordinates

Image coordinates (u, v) are a two dimensional coordinate system with units of *pixels* which denote positions in a video image. The origin of the system is at the top left corner of the video image. The u dimension increases to the right and the v dimension increases downwards. The r coordinate in the (u, v, r) triplet is an optional dimension representing the range in metres from the camera to that pixel. velocities in image space are measured in pixels/second, and in range: metres/second

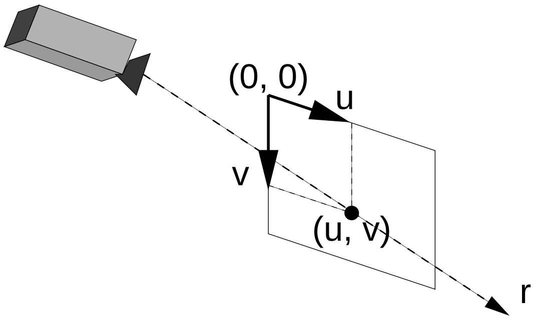

Figure 3: *Definition of image coordinates*

4.2 Pan/tilt sperical coordinates

This coordinate system is designed specifically for pan/tilt units. To define this coordinate system, consider a pan/tilt device mounted horizontally with a sensor placed on top of it. The origin of this coordinate system is defined as the camera looking “straight ahead” from the pan/tilt unit, and being completely horizontal. The pan angle θ_{pan} represents a rotation about the “pan axis” of the pan tilt unit. This angle is measured as positive in the anitclockwise direction when viewed from the top. The tilt angle ϕ_{tilt} represents a rotation about “tilt axis” of the pan tilt unit. This angle is measured as positive in the direction which makes the camera point “upwards”. The twist angle ρ_{twist} represents a rotation about the optical axis of the camera. This angle is measured as positive in the anticlockwise direction looking towards the “front” of the camera. The units of this coordinate system are either degrees or radians. All angular velocities are measured in rads/second.

The range coordinate r is measured in metres and taken in the direction of the optical axis of the camera.

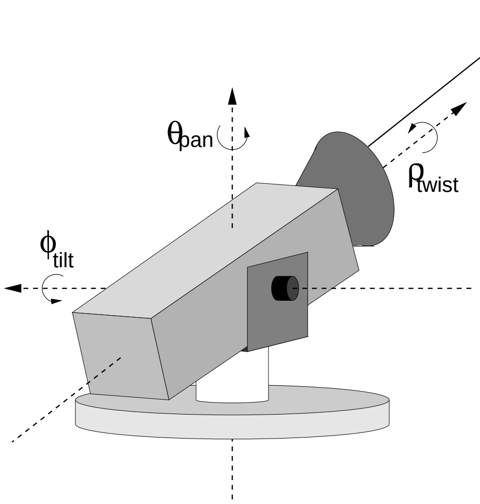

Figure 4: *Definition of pan/tilt coordinates*

The relationship between these rotational dimensions, and the general rotational dimensions of the global coordinate systems (LVCS and GPS), as described in section 4.5.

4.3 LVCS coordinates

An LVCS is a Cartesian coordinate system with its origin located somewhere at or near the site of interest. Point coordinates are represented as x,y,z distances, in metres, from the origin, where the coordinate system is oriented so that the positive X axis points true East, positive Y points true North, and positive Z points up (antiparallel to gravity). All that is needed to completely specify the LVCS is the 3D geodetic (GPS) coordinate of the origin point. For a description of the rotational dimensions (θ, ϕ, ρ) see section 4.5. Linear velocities in LVCS are all measured in units of metres/second

4.4 GPS coordinates

GPS coordinates are defined with respect to the WGS84 ellipsoid, and are represented by three values: latitude, longitude, and altitude in meters above the ellipsoid. Latitude and longitude are represented either as a single, floating point number denoting degrees, or as three separate numbers representing integer degrees, integer minutes, and floating point seconds. Latitude ranges from -90.0 to +90.0 degrees, with positive numbers representing latitudes North of the equator, and negative numbers denoting Southern latitudes. Longitude ranges from -180.0 to 180.0, with negative numbers meaning West of the Central Meridian), and positive numbers meaning East. The alternative degrees, minutes, second notation is supported because many instrument readouts give results in that format. For a description of the rotational dimensions (θ, ϕ, ρ) see section 4.5. Velocities in Lat and Lon are measured in units of rads/sec, and velocities in Alt are measured in metres/sec.

4.5 Rotational dimensions

Both the LVCS and GPS coordinate systems use the same definition of rotational dimensions (θ, ϕ, ρ). θ is defined as a rotation about the z or Alt axis. Positive θ is anticlockwise when viewed from above. ϕ is defined as a rotation about the y or Lon axis. It is defined as positive in the "upwards" direction. And ρ is defined as a rotation about the x or Lat axis, with positive values defined as an anticlockwise rotation when "looking" from the origin. All angular velocities are measured in rads/second.

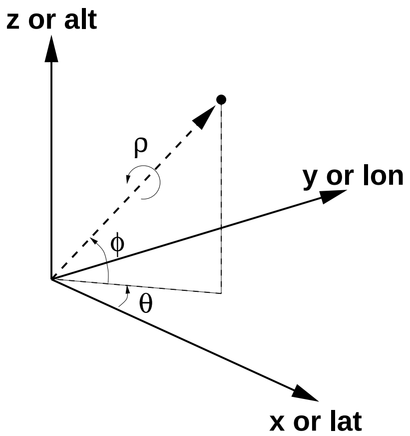

Figure 5: *Definition of rotational dimensions*

It is important to note that the operations of rotations about various axes is not commutative, consequently it is important to define the order in which these rotations are executed. In this implementation, the order of execution of rotations is

$$\theta \rightarrow \phi \rightarrow \rho \rightarrow \theta_{\text{pan}} \rightarrow \phi_{\text{tilt}} \rightarrow \rho_{\text{twist}}$$

so to determine the orientation of a unit vector $\vec{x}$ under a general rotation and pan/tilt, the following transformation must be applied

$$\vec{x}' = R_{\rho_{\text{twist}}} R_{\phi_{\text{tilt}}} R_{\theta_{\text{pan}}} R_\rho R_\phi R_\theta \vec{x}$$

where R_α is the rotation matrix associated with the dimension α.

4.6 time

All coordinate systems and entities in this architecture use the same definition of time, which is GPS time.

5 Supported data representations

In this implementation, five subgroups of data can be transfered via the communications protocol: commands, sensor data, imagery data, target descriptions, and event descriptions.

5.1 Command information

Command information
ID
Type
Relevant target IDs
Relevent event IDs

Table 2: *Overview of available command information*

5.1.1 Command ID

The command ID is a unique identifier which is used for logging command and control traffic. Each VSAM node will keep a running count of commands is has sent and will use this count as the identifier for all commands.

5.1.2 Command type

The command type is the actual command which is contained in the message. For a list of commands, see appendix A.

5.1.3 Target and event IDs

These are the the targets of events about which commands refer. For example, the OCU can task a sensor to *"search for target T1 at position (x, y, z)"* or *"Look for a rendezvous (event) between targets T3 and T5"*

5.2 Sensor information

5.2.1 Sensor ID

The sensor ID is a unique identifier per SPU agreed upon prior to run time. The identifiers which will be supported in the 1998 VSAM demo are shown in appendix B.

Sensor data
ID
Type
Status
Text label
Platform position, uncertainty and velocity
Platform pose, uncertainty and velocity
Sensor internal calibration
Sensor visual footprint
Sensor total field of regard

Table 3: *Overview of available sensor information*

5.2.2 Sensor type

The sensor type is an descriptor of the modality of the sensor. A list of currently supported modalities is provided in appendix B.

5.2.3 Sensor status

The status of a sensor describes what it is currently doing. A list of currently supported values for status is provided in appendix B

5.2.4 Text label

The text label for sensors is currently unused, but in future, will allow sensors to join the network and provide text descriptions of themselves.

5.2.5 Platform position & pose

This section of data contains the position and pose of the sensor platform. The currently supported features are:

- Sensor platform location in LVCS coordinates (x, y, z) or GPS coordinates (Lat, Lon, Alt), as well as uncertainties these quantities (Δx, Δy, Δz) and (ΔLat, ΔLon, ΔAlt).
- Sensor platform velocities in both LVCS and GPS coordinates ($\dot{x}$, $\dot{y}$, $\dot{z}$) and ($\dot{\mathrm{Lat}}$, $\dot{\mathrm{Lon}}$, $\dot{\mathrm{Alt}}$).
- Camera mount orientation (**NOT PTU position**) (θ, ϕ, ρ) and uncertainty in orientation ($\Delta\theta$, $\Delta\phi$, $\Delta\rho$).
- Camera mount angular velocity (**NOT PTU speed**) ($\dot{\theta}$, $\dot{\phi}$, $\dot{\rho}$)
- Pan/tilt/zoom values (θ_{pan}, ϕ_{tilt}, ρ_{twist}, Zoom) and velocities ($\dot{\theta}_{\mathrm{pan}}$, $\dot{\phi}_{\mathrm{tilt}}$, $\dot{\rho}_{\mathrm{twist}}$, $\dot{\mathrm{Zoom}}$). Zoom velocity is measured in "zoom units"/second, where encoding of zoom parameters is camera dependent. IFD sensors encode zoom as a known non-linear function mapping $0 \rightarrow 1023$ onto $1\times \rightarrow 11\times$.
- Pan/tilt/zoom region of interest. This is represented as a polygon in PTZ space (θ^1_{pan}, ϕ^1_{tilt}, ρ^1_{twist}, Zoom^1) $\rightarrow$ (θ^N_{pan}, ϕ^N_{tilt}, ρ^N_{twist}, Zoom^N), or a polygon in LVCS space (x^1, y^1, z^1) $\rightarrow$ (x^N, y^N, z^N). This can be used, for example, to task a sensor to *"search for target T in region R"*.

5.2.6 Sensor internal calibration information

The following information about sensor internam parameters and image formation is supported:

- Image size in pixels (U_size, V_size)
- Horizontal and vertical camera fields of view (H_fov, V_fov) in degrees or radians
- Camera internal parameters Focus, Iris, Image centre (U_c, V_c) in pixels, and constant of radial distortion (κ). Units of Iris and Focus is TBD.

5.2.7 Sensor visual footprint information

Sensors can represent and transmit their current "footprint" information which describes the viewing volume of the camera. Footprints can be described in one of two ways:

- A "bounding box" in pan/tilt space (θ^1_{pan}, ϕ^1_{tilt}) $\rightarrow$ (θ^2_{pan}, ϕ^2_{tilt}) where "1" represents the top left corner of the bounding box and "2" represents the bottom right corner of the bounding box, or
- A polygon on the ground in LVCS coordinates $(x, y, z)^1 \rightarrow (x, y, z)^N$ where N is the number of vertices.

5.2.8 Sensor field of regard (FOR) or limit information

Sensors can also represent and transmit their total field of regard information in a similar manner to their footprint data. The field of regard is the total area which is potentially visible to the sensor or can be considered the sensor's operational limits. These can also be described in one of two ways:

- The sensor limits in pan/tilt space (θ^1_{pan}, ϕ^1_{tilt}) $\rightarrow$ (θ^2_{pan}, ϕ^2_{tilt}) where "1" represents the "top left" corner of the sensor limits and "2" represents the "bottom right" corner of the sensor limits, or
- A polygon on the ground in LVCS coordinates $(x, y, z)^1 \rightarrow (x, y, z)^N$ where N is the number of vertices.

5.3 Imagery information

Imagery data
ID
Type
Text label
Still imagery data
Video clip data

Table 4: *Overview of available imagery information*

5.3.1 Imagery ID information

Imagery IDs are identifiers for each imagery datum. They are assigned by the originating VSAM node.

5.3.2 Imagery type information

CURRENTLY UNSUPPORTED

5.3.3 Imagery text label

This field is designed to apply text labels to imagery.

5.3.4 Still imagery data

Still images can be transmitted via this protocol. The supported image types are given in appendix C.

5.3.5 Video imagery data

Not supported in 1998. Video clips can be transmitted via this protocol. The envisaged video types are given in appendix C.

5.4 Target information

Target data
ID
Type
Text label
Target position and velocity
Target description
Target trajectory

Table 5: *Overview of available target information*

5.4.1 Target ID information

Target IDs are unique identifiers for each target the system knows about. When a new target is identified by a SPU, it requests an ID and the OCU either identifies it as a previously viewed target, or assigns a new value to it.

5.4.2 Target type information

Target types describe the classification of a particular target. The types which are currently supported are given in appendix D.

5.4.3 Target text label

This field is designed to apply text labels to targets if they can be individually identified.

5.4.4 Target position and velocity

The target's position can be specified in either image coordinate (u, v, r) where r is range (if accessible) or LVCS coordinates (x, y, z). The covariance matrix of the position in LVCS coordinates can also be represented. Velocities are also specified in either image coordinates ($\dot{u}$, $\dot{v}$, $\dot{r}$) or LVCS coordinates ($\dot{x}$, $\dot{y}$, $\dot{z}$).

5.4.5 Target description information

Targets can be described using a number of image based metrics. The supported descriptions are:

- Size: the number of pixels on target

- Bounding box: the bounding box of the target's image in image coordinates $(u^1, v^1) \rightarrow (u^2, v^2)$ where "1" denotes the "top left" corner of the bounding box and "2" denotes the "bottom right" corner.
- Skeleton: the "star" skeleton of the target can be represented in image coordinates by the centroid of the target (u_c, v_c), followed by a list of "extremal" points $(u_1, v_1) \rightarrow (u_N, v_N)$
- Boundary: the target's boundry is represented as a series of 8-connected pixels in image coordinates describing the boundary of the target $(u_1, v_1) \rightarrow (u_N, v_N)$
- Colour component histogram: Colour histograms of target information are representable at variable resolutions, and in variable colour spaces. The supported colour spaces are shown in appendix D
- Image template: a depiction of the target can be represented as an image template of arbitrary size. Several different pixel packing schemes are currently supported - see appendix D.
- Depth map: if sensors are capable of providing depth maps for targets, they can be represented as depth images along with the appropriate disparity data.
- Behaviour: there are also several pre-defined behaviours which can be used to describe a target. These are listed in appendix D.

5.4.6 Target trajectories

A "trajectory" in this protocol is defined as a time stamped list of any target property. This is a more general meaning than the traditional one. While it is allowable to represent trajectories as a time stamped series of target positions (which is the traditional definition), it is also possible to represent a "trajectory" of any of the target descriptors from section 5.4.5. For example, a "trajectory" of image templates would result in a small animation. Or a "trajectory" of behaviours could be used to detect anomalous activities such as a mugging.

5.5 Event information

Event data
ID
Type
Text label
Event position
Target list

Table 6: *Overview of available event information*

5.5.1 Event ID

Events have unique identifiers specified by the originating entity.

5.5.2 Event types

Event types describe what is taking place. A list of event types can be found in appendix E.

5.5.3 Target list information

Events can occur between targets - for example, *"a rendezvous (event) occurs between Bob (target) and Alan (target)"*. Relevant target IDs can be associated with specific events.

6 The VSAM network

This section describes how to physically connect to the VSAM network, and the network traffic guidelines to ensure that the network is not overloaded.

6.1 Physical connexion to the VSAM network

All FREs wishing to provide an SPU or visualisation tool will be able to access VSAM network. A number of standard 10base-T ports with RJ-45 connexion jacks will be made available to FREs from a central IFD testbed Ethernet hub.

FREs will be responsible for providing their own Ethernet cables and connectors, or other Ethernet transmission system. Space at the operator control station may be limited, so FREs should expect their equipment to be positioned up to 25 feet distant from the IFD testbed Ethernet hub.

6.2 Internet protocols

In this implementation, the IP address and host name of the OCU will be made available and it will be the responsibility of individual FREs to open and maintain appropriate communication channels with the OCU.

Two types of third party VSAM nodes will be supported in this implementation: FRE SPUs and FRE Visualisation (VIS) nodes. These two types of entity will use different packet structures and internet protocols. SPUs will connect to the OCU via a TCP/IP socket and will communicate using the Carnegie Mellon Packet Architecture (CMUPA), and the VIS nodes will connect using the UDP/IP and communicate using the DIS packet protocol (described in detail in the DIS specification document).

VSAM node	Internet protocol	packet structure
SPU	TCP/IP	CMUPA
VIS	UDP/IP	DIS

Table 7: *Internet protocols for VSAM nodes*

6.3 Network communication details

Once the SPU succeeds in establishing a connection with the OCU, handshaking shall take place. The handshaking process will identify the SPU and report its capabilities to the OCU, and OCU may limit the resource usage of the SPU.

6.4 Network traffic guidelines

The protocol described in this document is very flexible and allows entities to send large packets at high frequencies. To prevent this from overloading the network, a series of guide-

SPU	OCU
Send CM_I_AM_ALIVE Send bitmask of supported data and initial sensor status	 Reply with CM_GEN_ACK or CM_RECV_ERROR command *Optional* Send CM_SET_BANDWIDTH and/or CM_SET_PACKET_DESC

Table 8: *Handshaking sequence between SPU and OCU*

lines are presented which, it is hoped, will be adhered to by third party users of the VSAM network.

6.4.1 OCU to SPU traffic guidelines

The following guidelines are for communications between the OCU and SPUs

- The OCU will send data and requests to SPUs in an asynchronous manner. There will be no guidelines to restrict traffic from the OCU to SPUs.
- SPUs should send any information requested by the OCU immediately upon receiving a request. The only exception to this guideline should be if the requested information is not available or supported.
- SPUs should send only new or incrementally updated data to the OCU. For example, if the position of the SPU does not change between packets, do not encode this information in a new data packet.
- SPUs should send only the minimum relevant data. Send only the information required to facilitate communication. For example, it may not be necessary to send imagery of a target with every packet. It may be sufficient to send only that target's ID and position.
- SPUs should restrict communication frequency to $\leq$ 4Hz. It is unlikely that events will be unfolding at a higher frequency than this. Note that if no new information is being obtained by an SPU, it is unnecessary to send any data.
- SPUs should assume unlimited bandwidth unless OCU specifies otherwise. Nevertheless, the 4Hz communication limit should be obeyed at all time, unless SPU is responding to urgent data request, which shall be fulfilled immediately.
- SPUs are permitted to send requests and commands to other SPUs, however this is discouraged for the 1998 VSAM network.
- When sending trajectories, SPUs should not send complete trajectories, but rather, incremental updates. The only information present should be that which has been obtained between packets.

6.4.2 OCU to VIS traffic guidelines

The following guidelines are for communications between the OCU and VIS nodes

- Communiation between VIS nodes and the OCU is unidirectional. It is not permitted for VIS nodes to broadcast information of their own.
- VIS nodes can expect data packets from the OCU at $\approx$ 1Hz.

6.5 Future considerations

In future implementation of the communication protocol, the operator will have the ability to adjust tracking algorithm parameters from the GUI.

7 CMUPA protocol specification

The complete specification of the Carnegie Mellon Packet architecture is contained in this section.

7.1 Byte ordering

All data types shall adhere to network bit ordering and byte ordering for the purposes of transmission. The leftmost bit of any byte is considered the most significant bit, and the leftmost byte of any word is considered the most significant.

Words are transmitted in order from Most Significant Byte to Least significant byte.

3	2	1	0	Byte order
7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	Bit order
1	2	3	4	Transmission order

Figure 6: *Words are ordered from right (LSB) to left (MSB). Bytes are ordered from right (LSB) ot left (MSB). Data is transmitted from left (MSB first) to right (LSB last).*

7.2 Data types

Two data types are supported under this protocol: integers and floating point numbers. All floating point numbers adhere to the IEEE floating point representation format.

Data type	Notation	description
Integer	`UCHAR`	unsigned 8-bit int
	`CHAR`	signed 2's-compliment 8-bit int
	`UINT16`	unsigned 16-bit int
	`INT16`	signed 2's-compliment 16-bit int
	`UINT32`	unsigned 32-bit int
	`INT32`	signed 2's-compliment 32-bit int
Floating point	`FLOAT32`	32-bit float IEEE 754-1985 standard

Table 9: *Supported data types*

7.3 Data representation

Various types of data require particular packing definitions. Note that some individual fields within the packet structure require special descriptions. These will be described in the following sections.

7.3.1 Enumerations

All enumerated data sets will be represented as `UINT16` data types, and consecutively numbered starting at zero.

7.3.2 Timestamps

All time measurements in the VSAM network are synchronised to Universal Coordinated Time (UTC - or GPS time). Timestamps are represented as `UINT32` data types whose value represents the number of milliseconds since the start of the current day (12 midnight).

7.3.3 Angles

All angles are assumed to have have values between -180.0 degrees and +179.$\dot{9}$ degrees (or $[-\pi, \pi)$ radians).For transmission, an angle is represented as an `INT32` data type whose value represents a normalised angle. The normalisation constant is the maximum positive value which can be represented by an `INT32` data type - which is `MAX_INT32 = 2 ** 31 - 1`. So an angle is represented as:

```
VALUE = (ANGLE/180.0) * MAX_INT32
```

for an angle measured in degrees.

7.3.4 Angular velocities

All angular velocities are represented as floating point numbers with units of rads/sec.

7.3.5 Bitmasks

Various fields are bitmasks in the CMUPA protocol. All such fields are represented as `UINT16` or `UINT32` data types. Bit fields are packed into these field from the least significant bit to most significant bit. Bit values of “0” (reset) mean that the relevant data is NOT present in the packet, and bit values of “1” (set) mean that the data IS present.

7.4 Packet structure

CMUPA packets have a header, followed by up to five optional data sections (command, sensor, imagery, target, and/or event), each containing one or more data blocks.

7.5 Packet headers

Field	Representation	Type	Size (bytes)
Packet size		`UINT32`	4
Packet ID		`UINT32`	4
Scenario ID		`UINT16`	2
Source entity ID	enumeration	`UINT16`	2
Destination entity ID	enumeration	`UINT16`	2
Time	timestamp	`UINT32`	4
H - Packet content bitmask	bitmask	`UINT16`	2

7.5.1 Packet size

Packet size, in bytes, of the entire packet including every subsection.

HEADER Bitmask H	
OPTIONAL	Number of command blocks
COMMAND	Command block 1
SECTION	⋮
(Bit H0)	Command block N
OPTIONAL	Number of sensor blocks
SENSOR	Sensor block 1
SECTION	⋮
(Bit H1)	Sensor block N
OPTIONAL	Number of imagery blocks
IMAGERY	Imagery block 1
SECTION	⋮
(Bit H2)	Imagery block N
OPTIONAL	Number of target blocks
TARGET	Target block 1
SECTION	⋮
(Bit H3)	Target block N
OPTIONAL	Number of event blocks
EVENT	Event block 1
SECTION	⋮
(Bit H4)	Event block N

Table 10: *CMUPA packet structure*

7.5.2 Packet ID

Each packet will have a unique ID provided by the originating entity.

7.5.3 Scenario ID

Each VSAM scenario will be given a unique scenario ID which will be maintained by the OCU.

7.5.4 Source entity ID

Each node in the VSAM network will have a unique ID. The originating entity will provide its own ID in this field. A list of currently supported IDs is provided in appendix F.

7.5.5 Destination entity ID

Each node in the VSAM network will have a unique ID. The originating entity will provide the ID of the intended recipient of the packet in this field. A list of currently supported IDs is provided in appendix F. Note that an SPU may direct packets to other SPUs or VIS nodes, but this practice is discouraged in the 1998 VSAM implementation. Note also, that some information cannot be transmitted to VIS nodes as they have a different packet structure which does not support all of the data types supported under the CMUPA.

7.5.6 H - Packet content bitmask

This bitmask describes which of the five data sections are physically present in the packet. The format of the bitmask is

Bit	H15	H14	H13	H12	H11	H10	H9	H8
Mask	—	—	—	—	—	—	—	—
Bit	H7	H6	H5	H4	H3	H2	H1	H0
Mask	—	—	—	Event	Target	Imagery	Sensor	Command

7.6 The command section

This section contains any command and control information, requests for data, etc.

Bitmask	Field	Representation	Type	Size (bytes)
H0	Number of command data blocks		`UINT16`	2
COMMAND 1 DATA BLOCK				

H0	Number of bytes
	Command ID
	Command type
	C - Command data bitmask
		
		enumeration
		bitmask
			`UINT16`
			`UINT16`
			`UINT32`
			`UINT32`
				2
				2
				4
				4

H0 & C0	**Optional** Target fields			
H0 & C1	**Optional** Event fields			
H0 & C2	**Optional** Packet bitmasks			
H0 & C3	**Optional** Bandwidth			
⋮				
COMMAND N DATA BLOCK				

7.6.1 Number of command data blocks

This field denotes the number of command data blocks which are present in the command section. This number and all subsequent command data will only be present in the packet if bit H0 is set in the header bitmask.

7.6.2 Number of bytes in a command block

This field contains the number of bytes in its own command data block, **NOT** the total command section.

7.6.3 Command ID

Each VSAM entity will keep a count of the commands they issue and this number will be used as a unique command identifier.

7.6.4 Command type

A list of the currently supported commands can be found in appendix A.

7.6.5 C - command data bitmask

This bitmask describes which data fields are physically present in a command data block.

Bit	C31	C30	C29	C28	C27	C26	C25	C24
Mask	—	—	—	—	—	—	—	—
Bit	C23	C22	C21	C20	C19	C18	C17	C16
Mask	—	—	—	—	—	—	—	—
Bit	C15	C14	C13	C12	C11	C10	C9	C8
Mask	—	—	—	—	—	—	—	—
Bit	C7	C6	C5	C4	C3	C2	C1	C0
Mask	—	—	—	—	Bandwidth	Packet bitmasks	Event fields	Target fields

7.6.6 Target fields in a command

If the C0 bit of the command data bitmask is set, these fields will be included in the command data block

Bitmask	Field	Representation	Type	Size (bytes)
H0 & C0	Number of targets		`UINT16`	2
	Target 1 ID		`UINT16`	2
	⋮			
	Target N ID		`UINT16`	2

7.6.7 Event fields in a command

If the C1 bit of the command data bitmask is set, these fields will be included in the command data block

Bitmask	Field	Representation	Type	Size (bytes)
H0 & C1	Number of events		`UINT16`	2
	Event 1 ID		`UINT16`	2
	⋮			
	Event N ID		`UINT16`	2

7.6.8 Packet bitmasks

If the C2 bit of the command data bitmask is set, these fields will be included in the command data block

Bitmask	Field	Representation	Type	Size (bytes)
H0 & C2	Command bitmask	bitmask	`UINT32`	4
	Sensor bitmask	bitmask	`UINT32`	4
	Imagery bitmask	bitmask	`UINT32`	4
	Target bitmask	bitmask	`UINT32`	4
	Trajectory bitmask	bitmask	`UINT32`	4
	Event bitmask	bitmask	`UINT32`	4

7.6.9 Bandwidth

If the C3 bit of the command data bitmask is set, this field will be included in the command data block

Bitmask	Field	Representation	Type	Size (bytes)
H0 & C3	Bandwidth Allocated	bytes per second	`UINT32`	4

7.7 The sensor section

This section contains any information directly pertinent to sensors and their platforms.

Bitmask	Field	Representation	Type	Size (bytes)
H1	Number of sensor data blocks		`UINT16`	2
SENSOR 1 DATA BLOCK				
H1	Number of bytes		`UINT16`	2
	Sensor ID	enumeration	`UINT16`	2
	Sensor type	enumeration	`UINT16`	2
	Sensor status	enumeration	`UINT16`	2
	Length of text label (N)		`UINT16`	2
	Text label		`UCHAR`	N
	S - Sensor data bitmask	bitmask	`UINT32`	4
H1 & S0	**Optional** LVCS position fields			
H1 & S1	**Optional** LVCS position uncertainty fields			
H1 & S2	**Optional** LVCS velocity fields			
H1 & S3	**Optional** GPS position fields			
H1 & S4	**Optional** GPS position uncertainty fields			
H1 & S5	**Optional** GPS velocity fields			
H1 & S6	**Optional** Sensor mount orientation fields			
H1 & S7	**Optional** Sensor mount orientation uncertainty fields			
H1 & S8	**Optional** Sensor mount orientation angular velocity fields			
H1 & S9	**Optional** Image size (pixels) fields			
H1 & S10	**Optional** Sensor FOV fields			
H1 & S11	**Optional** Sensor internal parameter fields			
H1 & S12	**Optional** Pan/tilt/zoom orientation fields			
H1 & S13	**Optional** Pan/tilt/zoom angular velocity fields			
H1 & S14	**Optional** region of interest in Pan/tilt/zoom coords fields			
H1 & S15	**Optional** region of interest in LVCS coords fields			
H1 & S16	**Optional** Pan/tilt/zoom sensor footprint fields			
H1 & S17	**Optional** LVCS sensor footprint fields			
H1 & S18	**Optional** Pan/tilt/zoom sensor limits fields			
H1 & S19	**Optional** LVCS sensor limits fields			
⋮				
SENSOR N DATA BLOCK				

7.7.1 Number of sensor data blocks

This field denotes the number of sensor data blocks which are present in the sensor section. This number and all subsequent sensor data will only be present in the packet if bit H1 is set in the header bitmask.

7.7.2 Number of bytes in a sensor block

This field contains the number of bytes in its own sensor data block, **NOT** the total sensor section.

7.7.3 Sensor ID

This enumeration denotes the individual ID of the sensor to which the subsequent data pertains. A list of sensor IDs can be found in appendix B

7.7.4 Sensor type

A list of the currently supported sensor types can be found in appendix B.

7.7.5 Sensor status

This field describes the current status of the sensor. A list of currently supported values for this field can be found in appendix B.

7.7.6 Sensor text label

This protocol allows an optional text label to be associated with a sensor. In general, this field will not be used in this implementation. It is envisaged that new sensors joining the network will provide a text description of themselves.

7.7.7 S - sensor data bitmask

This bitmask describes which data fields are physically present in a sensor data block.

Bit	S31	S30	S29	S28	S27	S26	S25	S24
Mask	—	—	—	—	—	—	—	—
Bit	S23	S22	S21	S20	S19	S18	S17	S16
Mask	—	—	—	—	lvcs lim	ptz lim	lvcs fp	ptz fp
Bit	S15	S14	S13	S12	S11	S10	S9	S8
Mask	lvcs roi	ptz roi	ptz vel.	ptz	Int pars	FOV	size	ang vel
Bit	S7	S6	S5	S4	S3	S2	S1	S0
Mask	Mt err	Mt pos	GPS vel	GPS err	GPS	lvcs vel	lvcs err	lvcs

7.7.8 LVCS sensor position fields (lvcs)

If the S0 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S0	LVCS X coordinate (metres)		`FLOAT32`	4
	LVCS Y coordinate (metres)		`FLOAT32`	4
	LVCS Z coordinate (metres)		`FLOAT32`	4

7.7.9 LVCS sensor position uncertainty fields (lvcs err)

If the S1 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S1	LVCS ΔX in metres		`FLOAT32`	4
	LVCS ΔY in metres		`FLOAT32`	4
	LVCS ΔZ in metres		`FLOAT32`	4

7.7.10 LVCS sensor linear velocity fields (lvcs vel)

If the S2 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S2	LVCS $\dot{X}$ in metres/sec		`FLOAT32`	4
	LVCS $\dot{Y}$ in metres/sec		`FLOAT32`	4
	LVCS $\dot{Z}$ in metres/sec		`FLOAT32`	4

7.7.11 GPS sensor position fields (GPS)

If the S3 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S3	GPS Lat	Angle	`INT32`	4
	GPS Lon	Angle	`INT32`	4
	GPS Alt		`FLOAT32`	4

7.7.12 GPS sensor position uncertainty fields (GPS err)

If the S4 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S4	GPS ΔLat	Angle	`INT32`	4
	GPS ΔLon	Angle	`INT32`	4
	GPS ΔAlt		`FLOAT32`	4

7.7.13 GPS sensor "linear" velocity fields (GPS vel)

If the S5 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S5	GPS $L\dot{a}t$	Angular vel.	`FLOAT32`	4
	GPS $L\dot{o}n$	Angular vel.	`FLOAT32`	4
	GPS $A\dot{l}t$		`FLOAT32`	4

7.7.14 Sensor mount orientation fields (Mt pos)

If the S6 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S6	θ	Angle	`INT32`	4
	ϕ	Angle	`INT32`	4
	ρ	Angle	`INT32`	4

7.7.15 Sensor mount orientation uncertainty fields (Mt err)

If the S7 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S7	$\Delta\theta$	Angle	`INT32`	4
	$\Delta\phi$	Angle	`INT32`	4
	$\Delta\rho$	Angle	`INT32`	4

7.7.16 Sensor mount angular velocity fields (Mt vel)

If the S8 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S8	$\dot{\theta}$	Angular vel.	FLOAT32	4
	$\dot{\phi}$	Angular vel.	FLOAT32	4
	$\dot{\rho}$	Angular vel.	FLOAT32	4

7.7.17 Image size (pixels) fields (size)

If the S9 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S9	Image width u (pixels)		UINT16	2
	Image height v (pixels)		UINT16	2

7.7.18 Sensor field of view fields (FOV)

If the S10 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S10	Horizontal FOV	Angle	INT32	4
	Vertical FOV	Angle	INT32	4

7.7.19 Sensor internal parameters fields (Int pars)

If the S11 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S11	Iris		INT16	2
	Focus		INT16	2
	Image centre u_c (pixels)		INT16	2
	Image centre v_c (pixels)		INT16	2
	Coeff. of radial distortion κ		FLOAT32	4

NOTE: the encoding of Iris and Focus are TBD

7.7.20 Pan/tilt/zoom position fields (ptz)

If the S12 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S12	θ_{pan}	Angle	INT32	4
	ϕ_{tilt}	Angle	INT32	4
	ρ_{twist}	Angle	INT32	4
	Zoom		FLOAT32	4

Note: encoding zoom is camera specific. In the case of IFD sensors, zoom is a known non-linear function where the values $0 \rightarrow 1023$ map onto zoom settings $1\times \rightarrow 11\times$.

7.7.21 Pan/tilt/zoom velocity fields (ptz vel)

If the S13 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S13	$\Delta\theta_{\text{pan}}$	Angular vel.	FLOAT32	4
	$\Delta\phi_{\text{tilt}}$	Angular vel.	FLOAT32	4
	$\Delta\rho_{\text{twist}}$	Angular vel.	FLOAT32	4
	ΔZoom		FLOAT32	4

7.7.22 Region of interest in P/T/Z coords. fields (ptz roi)

If the S14 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S14	Number of ptz positions in ROI		INT16	2
	θ^1_{pan}	Angle	INT32	4
	ϕ^1_{tilt}	Angle	INT32	4
	ρ^1_{twist}	Angle	INT32	4
	Zoom^1		FLOAT32	4
	$\vdots$			
	θ^N_{pan}	Angle	INT32	4
	ϕ^N_{tilt}	Angle	INT32	4
	ρ^N_{twist}	Angle	INT32	4
	Zoom^N		FLOAT32	4

7.7.23 Region of interest in LVCS coords. fields (lvcs roi)

If the S15 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S15	Number of LVCS positions in ROI		INT16	2
	LVCS X^1 coord (metres)		FLOAT32	4
	LVCS Y^1 coord (metres)		FLOAT32	4
	LVCS Z^1 coord (metres)		FLOAT32	4
	$\vdots$			
	LVCS X^N coord (metres)		FLOAT32	4
	LVCS Y^N coord (metres)		FLOAT32	4
	LVCS Z^N coord (metres)		FLOAT32	4

7.7.24 Sensor footprint in pan/tilt coords fields (ptz fp)

If the S16 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S16	Top left θ_{pan}	Angle.	INT32	4
	Top left ϕ_{tilt}	Angle	INT32	4
	Bottom right θ_{pan}	Angle.	INT32	4
	Bottom right ϕ_{tilt}	Angle	INT32	4

7.7.25 Sensor footprint in LVCS coords. fields (lvcs fp)

If the S17 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S17	No. of LVCS coords in footprint		`INT16`	2
	LVCS X^1 coord (metres)		`FLOAT32`	4
	LVCS Y^1 coord (metres)		`FLOAT32`	4
	LVCS Z^1 coord (metres)		`FLOAT32`	4
	$\vdots$			
	LVCS X^N coord (metres)		`FLOAT32`	4
	LVCS Y^N coord (metres)		`FLOAT32`	4
	LVCS Z^N coord (metres)		`FLOAT32`	4

7.7.26 Sensor limits in pan/tilt coords fields (ptz lim)

If the S18 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S18	Top left θ_{pan}	Angle.	`INT32`	4
	Top left ϕ_{tilt}	Angle	`INT32`	4
	Bottom right θ_{pan}	Angle.	`INT32`	4
	Bottom right ϕ_{tilt}	Angle	`INT32`	4

7.7.27 Sensor limits in LVCS coords. fields (lvcs lim)

If the S19 bit of the sensor data bitmask is set, these fields will be included in the sensor data block

Bitmask	Field	Representation	Type	Size (bytes)
H1 & S19	No. of LVCS coords. in limit		`INT16`	2
	LVCS X^1 coord (metres)		`FLOAT32`	4
	LVCS Y^1 coord (metres)		`FLOAT32`	4
	LVCS Z^1 coord (metres)		`FLOAT32`	4
	$\vdots$			
	LVCS X^N coord (metres)		`FLOAT32`	4
	LVCS Y^N coord (metres)		`FLOAT32`	4
	LVCS Z^N coord (metres)		`FLOAT32`	4

7.8 The imagery section

This section contains any imagery data (not specifically associated with targets - which could be contained in the target section).

Bitmask	Field	Representation	Type	Size (bytes)
H2	Number of Imagery data blocks		UINT16	2
IMAGERY 1 DATA BLOCK				
H2	Number of bytes		UINT32	4
	Imagery ID		UINT16	2
	Imagery type	enumeration	UINT16	2
	Length of text label (N)		UINT16	2
	Text label		UCHAR	N
	I - Imagery data bitmask	bitmask	UINT32	4
H2 & I0	**Optional** Still imagery fields			
H2 & I1	**Optional** Video clip fields			
⋮				
IMAGERY N DATA BLOCK				

7.8.1 Number of imagery data blocks

This field denotes the number of imagery data blocks which are present in the imagery section. This number and all subsequent imagery data will only be present in the packet if bit H2 is set in the header bitmask.

7.8.2 Number of bytes in an imagery block

This field contains the number of bytes in its own imagery data block, **NOT** the total imagery section.

7.8.3 Imagery ID

Each VSAM entity will keep a count of the imagery data they issue and this number will be used as a unique imagery identifier.

7.8.4 Imagery type

NOT CURRENTLY SUPPORTED

7.8.5 I - Imagery data bitmask

This bitmask describes which data fields are physically present in an imagery data block.

Bit	I31	I30	I29	I28	I27	I26	I25	I24
Mask	—	—	—	—	—	—	—	—
Bit	I23	I22	I21	I20	I19	I18	I17	I16
Mask	—	—	—	—	—	—	—	—
Bit	I15	I14	I13	I12	I11	I10	I9	I8
Mask	—	—	—	—	—	—	—	—
Bit	I7	I6	I5	I4	I3	I2	I1	I0
Mask	—	—	—	—	—	—	video	still

7.8.6 Still imagery fields

If the I0 bit of the imagery data bitmask is set, these fields will be included in the imagery data block

Bitmask	Field	Representation	Type	Size (bytes)
H2 & I0	Image time	timestamp	`UINT32`	4
	Image width u (pixels)		`UINT16`	2
	Image height v (pixels)		`UINT16`	2
	Image format	enumeration	`UINT16`	2
	Length of image data		`UINT32`	4
	Image data		`UCHAR`	??

Note: image format types can be found in appendix C

7.8.7 Video clip fields

If the I1 bit of the imagery data bitmask is set, these fields will be included in the imagery data block

Bitmask	Field	Representation	Type	Size (bytes)
H2 & I1	Start time	timestamp	`UINT32`	4
	End time	timestamp	`UINT32`	4
	Video width u (pixels)		`UINT16`	2
	Video height v (pixels)		`UINT16`	2
	Video format	enumeration	`UINT16`	2
	Length of video data		`UINT32`	4
	Video data		`UCHAR`	??

Note: video format types can be found in appendix C

7.9 The target section

This section contains any data relevant to specific targets, including their position, representation and trajectory.

Bitmask	Field	Representation	Type	Bytes
H3	Number of Target data blocks		UINT16	2
	TARGET 1 DATA BLOCK			
H3	Number of bytes		UINT16	2
	Target ID		UINT16	2
	Target type	enumeration	UINT16	2
	Length of text label (N)		UINT16	2
	Text label		UCHAR	N
	R - Target rep. bitmask	bitmask	UINT32	4
	T - Target trajectory bitmask	bitmask	UINT32	4
H3 & R0	**Optional** Target position in image coords fields			
H3 & R1	**Optional** Target position in LVCS fields			
H3 & R2	**Optional** Target pos. in LVCS covariance matrix fields			
H3 & R3	**Optional** Target velocity in image coords fields			
H3 & R4	**Optional** Target velocity in LVCS fields			
H3 & R5	**Optional** Target bounding box fields			
H3 & R6	**Optional** Target skeleton fields			
H3 & R7	**Optional** Target boundary fields			
H3 & R8	**Optional** Target size (pix on target) fields			
H3 & R9	**Optional** Target colour histogram fields			
H3 & R10	**Optional** Target image template fields			
H3 & R11	**Optional** Target depth map fields			
H3 & R12	**Optional** Target Depth map resolution fields			
H3 & R13	**Optional** Target behaviour fields			
H3 & T0	**Optional** Trajectory on/off length/timestamp fields			
H3 & T0 & T1	**Optional** Trajectory of position in image coords fields			
H3 & T0 & T2	**Optional** Trajectory of pos. in LVCS fields			
H3 & T0 & T3	**Optional** Trajectory of pos. in LVCS covariance matrix fields			
H3 & T0 & T4	**Optional** Trajectory of bounding box fields			
H3 & T0 & T5	**Optional** Trajectory of skeleton fields			
H3 & T0 & T6	**Optional** Trajectory of boundary fields			
H3 & T0 & T7	**Optional** Trajectory of size fields			
H3 & T0 & T8	**Optional** Trajectory of colour histogram fields			
H3 & T0 & T9	**Optional** Trajectory of image template fields			
H3 & T0 & T10	**Optional** Trajectory of depth map fields			
H3 & T0 & T11	**Optional** Trajectory of behaviour fields			
	⋮			
	TARGET N DATA BLOCK			

7.9.1 Number of target data blocks

This field denotes the number of target data blocks which are present in the target section. This number and all subsequent target data will only be present in the packet if bit H3 is set in the header bitmask.

7.9.2 Number of bytes in a target block

This field contains the number of bytes in its own target data block, **NOT** the total target section.

7.9.3 Target ID

The OCU will keep a master list of known targets, each with its own unique identifier. If an SPU detects a target for which it does not know the ID, it will request clarification from the OCU which will respond with an appropriate target ID.

7.9.4 Target type

Currently known target types are listed in appendix D.

7.9.5 Target text label

This protocol allows an optional text label to be associated with a sensor. In general, this field will not be used in this implementation. If specific targets can be identified, then associated labels may be transmitted with target information.

7.9.6 R - Target representation data bitmask

Target data blocks can support two types of target information: target representation information and target trajectory information. Target representation includes data types which describe the position, shape and appearance of a target. A "trajectory" is created by taking a number of timestamped instances of a representation type and joining them together.

The target representation data bitmask R describes which target description fields are physically present in a target data block.

Bit	R31	R30	R29	R28	R27	R26	R25	R24
Mask	—	—	—	—	—	—	—	—
Bit	R23	R22	R21	R20	R19	R18	R17	R16
Mask	—	—	—	—	—	—	—	—
Bit	R15	R14	R13	R12	R11	R10	R9	R8
Mask	—	—	behav	depth res	depth	template	histgm	size
Bit	R7	R6	R5	R4	R3	R2	R1	R0
Mask	bound	skel	b-box	lvcs vel	img vel	lvcs err	lvcs pos	img pos

7.9.7 T - Target trajectory data bitmask

Target data blocks can support two types of target information: target representation information and target trajectory information. Target representation includes data types which describe the position, shape and appearance of a target. A "trajectory" is created by taking a number of timestamped instances of a representation type and joining them together.

The target representation data bitmask T describes which trajectory description fields are physically present in a target data block.

Bit	T31	T30	T29	T28	T27	T26	T25	T24
Mask	—	—	—	—	—	—	—	—
Bit	T23	T22	T21	T20	T19	T18	T17	T16
Mask	—	—	—	—	—	—	—	—
Bit	T15	T14	T13	T12	T11	T10	T9	T8
Mask	—	—	—	—	behav	depth	template	histgm
Bit	T7	T6	T5	T4	T3	T2	T1	T0
Mask	size	bound	skel	b-box	lvcs err	lvcs pos	img pos	on/off

Note: if trajectories are used, bit T0 MUST BE SET

7.9.8 Target position in image coords fields (img pos)

If the R0 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R0	u (pixels)		`UINT16`	2
	v (pixels)		`UINT16`	2

7.9.9 Target position in LVCS coords fields (lvcs pos)

If the R1 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R1	x (metres)		`FLOAT32`	4
	y (metres)		`FLOAT32`	4
	z (metres)		`FLOAT32`	4

7.9.10 Target covariance matrix in LVCS coords fields (lvcs err)

If the R2 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R2	C_{00}		`FLOAT32`	4
	C_{01}		`FLOAT32`	4
	C_{02}		`FLOAT32`	4
	C_{10}		`FLOAT32`	4
	C_{11}		`FLOAT32`	4
	C_{12}		`FLOAT32`	4
	C_{20}		`FLOAT32`	4
	C_{21}		`FLOAT32`	4
	C_{22}		`FLOAT32`	4

where a covariance matrix is encoded

$$C = \begin{bmatrix} C_{00} & C_{01} & C_{02} \\ C_{10} & C_{11} & C_{12} \\ C_{20} & C_{21} & C_{22} \end{bmatrix}$$

7.9.11 Target velocity in image coords fields (img vel)

If the R3 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R3	$\dot{u}$ (pixels/sec)		`FLOAT32`	4
	$\dot{v}$ (pixels/sec)		`FLOAT32`	4

7.9.12 Target velocity in LVCS coords fields (lvcs vel)

If the R4 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R4	$\dot{x}$ (metres/sec)		`FLOAT32`	4
	$\dot{y}$ (metres/sec)		`FLOAT32`	4
	$\dot{z}$ (metres/sec)		`FLOAT32`	4

7.9.13 Target bounding box fields (b-box)

If the R5 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R5	Top left u (pixels)		`UINT16`	2
	Top left v (pixels)		`UINT16`	2
	Bottom right u (pixels)		`UINT16`	2
	Bottom right v (pixels)		`UINT16`	2

7.9.14 Target skeleton fields (skel)

If the R6 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R6	Centroid u (pixels)		`UINT16`	2
	Centroid v (pixels)		`UINT16`	2
	Number of extremal points		`UINT16`	2
	Extremal point 1 u (pixels)		`UINT16`	2
	Extremal point 1 v (pixels)		`UINT16`	2
	$\vdots$			
	Extremal point N u (pixels)		`UINT16`	2
	Extremal point N v (pixels)		`UINT16`	2

7.9.15 Target boundary fields (bound)

If the R7 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R7	Number of boundary pixels		`UINT16`	2
	Boundary point 1 u (pixels)		`UINT16`	2
	Boundary point 1 v (pixels)		`UINT16`	2
	$\vdots$			
	Boundary point N u (pixels)		`UINT16`	2
	Boundary point N v (pixels)		`UINT16`	2

7.9.16 Target size (pixels on target) fields (size)

If the R8 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R8	size (number of pixels)		`UINT16`	2

7.9.17 Target colour histogram fields (histgm)

If the R9 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R9	Colour space	enumeration	`UINT16`	2
	Number of channel 1 bins (N_1)		`UINT16`	2
	Number of channel 2 bins (N_2)		`UINT16`	2
	Number of channel 3 bins (N_3)		`UINT16`	2
	Channel 1 bin 1		`UINT16`	2
	⋮			
	Channel 1 bin N_1		`UINT16`	2
	Channel 2 bin 1		`UINT16`	2
	⋮			
	Channel 2 bin N_2		`UINT16`	2
	Channel 3 bin 1		`UINT16`	2
	⋮			
	Channel 3 bin N_3		`UINT16`	2

If only a monochrome histogram is required, use the first data channel. The list of supported colour spaces can be found in appendix D.

7.9.18 Target image template fields (template)

If the R10 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R10	Template width u (pixels)		`UINT16`	2
	Template height v (pixels)		`UINT16`	2
	Template pixel packing	enumeration	`UINT16`	2
	Template data		`UCHAR`	$u \times v$

Currently supported pixel packing values can be found in appendix D.

7.9.19 Target depth map fields (depth)

If the R11 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R11	Depth map width u (pixels)		`UINT16`	2
	Depth map height v (pixels)		`UINT16`	2
	Depth map pixel packing	enumeration	`UINT16`	2
	Depth map data		`UCHAR`	$u \times v$

Currently supported pixel packing values can be found in appendix D.

7.9.20 Target depth map resolution fields (depth res)

If the R12 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R12	Number of disparities (N)		`UINT16`	2
	Resolution lookup table		`FLOAT32`	$4 \times N$

7.9.21 Target behaviour fields (behav)

If the R13 bit of the target representation bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & R13	Behaviour	enumeration	`UINT16`	2

A list of current supported behaviour types is given in appendix D.

7.9.22 Trajectory on/off length/timestamp fields (on/off)

If the T0 bit of the target trajectory bitmask is set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & T0	Length of trajectory		`UINT16`	2
	Time 1 (first record)	timestamp	`UINT32`	4
	⋮			
	Time N (last record)	timestamp	`UINT32`	4

Note that timestamps must be monotonically increasing from time 1 to time N.

7.9.23 Trajectory of position in image coords fields (img pos)

If bits T0 and T1 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & T0 & T1	Position 1 u (pixels)		`UINT16`	2
	Position 1 v (pixels)		`UINT16`	2
	⋮			
	Position N u (pixels)		`UINT16`	2
	Position N v (pixels)		`UINT16`	2

7.9.24 Trajectory of position in LVCS coords fields (lvcs pos)

If bits T0 and T2 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & T0 & T2	Position 1 x (metres)		FLOAT32	4
	Position 1 y (metres)		FLOAT32	4
	Position 1 z (metres)		FLOAT32	4
	$\vdots$			
	Position N x (metres)		FLOAT32	4
	Position N y (metres)		FLOAT32	4
	Position N z (metres)		FLOAT32	4

7.9.25 Trajectory of covariance matrix in LVCS coords fields (lvcs err)

If bits T0 and T3 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & T0 & T3	Position 1 C_{00}		FLOAT32	4
	Position 1 C_{01}		FLOAT32	4
	Position 1 C_{02}		FLOAT32	4
	Position 1 C_{10}		FLOAT32	4
	Position 1 C_{11}		FLOAT32	4
	Position 1 C_{12}		FLOAT32	4
	Position 1 C_{20}		FLOAT32	4
	Position 1 C_{21}		FLOAT32	4
	Position 1 C_{22}		FLOAT32	4
	$\vdots$			
	Position N C_{00}		FLOAT32	4
	Position N C_{01}		FLOAT32	4
	Position N C_{02}		FLOAT32	4
	Position N C_{10}		FLOAT32	4
	Position N C_{11}		FLOAT32	4
	Position N C_{12}		FLOAT32	4
	Position N C_{20}		FLOAT32	4
	Position N C_{21}		FLOAT32	4
	Position N C_{22}		FLOAT32	4

where a covariance matrix is encoded

$$C = \begin{bmatrix} C_{00} & C_{01} & C_{02} \\ C_{10} & C_{11} & C_{12} \\ C_{20} & C_{21} & C_{22} \end{bmatrix}$$

7.9.26 Trajectory of bounding box fields (b-box)

If bits T0 and T4 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & T0 & T4	Top left 1 u (pixels)		`UINT16`	2
	Top left 1 v (pixels)		`UINT16`	2
	Bottom right 1 u (pixels)		`UINT16`	2
	Bottom right 1 v (pixels)		`UINT16`	2
	⋮			
	Top left N u (pixels)		`UINT16`	2
	Top left N v (pixels)		`UINT16`	2
	Bottom right N u (pixels)		`UINT16`	2
	Bottom right N v (pixels)		`UINT16`	2

7.9.27 Trajectory of skeleton fields (skel)

If bits T0 and T5 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Rep.	Type	Size (bytes)
H3 & T0 & T5	Centroid 1 u (pixels)		`UINT16`	2
	Centroid 1 v (pixels)		`UINT16`	2
	Skel. 1 No. of ext. points (S1)		`UINT16`	2
	Skel. 1 ext. point 1 u (pixels)		`UINT16`	2
	Skel. 1 ext. point 1 v (pixels)		`UINT16`	2
	⋮			
	Skel. 1 ext. point S1 u (pixels)		`UINT16`	2
	Skel. 1 ext. point S1 v (pixels)		`UINT16`	2
	⋮			
	⋮			
	Centroid N u (pixels)		`UINT16`	2
	Centroid N v (pixels)		`UINT16`	2
	Skel. N No. of ext. points (SN)		`UINT16`	2
	Skel. N ext. point 1 u (pixels)		`UINT16`	2
	Skel. N ext. point 1 v (pixels)		`UINT16`	2
	⋮			
	Skel. N ext. point SN u (pixels)		`UINT16`	2
	Skel. N ext. point SN v (pixels)		`UINT16`	2

7.9.28 Trajectory of target boundary fields (bound)

If bits T0 and T6 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Rep.	Type	Size (bytes)
H3 & T0 & T6	Boundary 1 number of points (B1)		`UINT16`	2
	Boundary 1 point 1 u (pixels)		`UINT16`	2
	Boundary 1 point 1 v (pixels)		`UINT16`	2
	$\vdots$			
	Boundary 1 point B1 u (pixels)		`UINT16`	2
	Boundary 1 point B1 v (pixels)		`UINT16`	2
	$\vdots$			
	$\vdots$			
	Boundary N number of points (BN)		`UINT16`	2
	Boundary N point 1 u (pixels)		`UINT16`	2
	Boundary N point 1 v (pixels)		`UINT16`	2
	$\vdots$			
	Boundary N point BN u (pixels)		`UINT16`	2
	Boundary N point BN v (pixels)		`UINT16`	2

7.9.29 Trajectory of target size fields (size)

If bits T0 and T7 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & T0 & T7	size 1 (number of pixels)		`UINT16`	2
	$\vdots$			
	size N (number of pixels)		`UINT16`	2

7.9.30 Trajectory of colour histograms fields (histgm)

If bits T0 and T8 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Rep.	Type	Size (bytes)
H3 & T0 & T8	Colour space 1	enumer.	`UINT16`	2
	Hist 1 No. of channel 1 bins ($H1_1$)		`UINT16`	2
	Hist 1 No. of channel 2 bins ($H1_2$)		`UINT16`	2
	Hist 1 No. of channel 3 bins ($H1_3$)		`UINT16`	2
	Hist 1 Channel 1 bin 1		`UINT16`	2
	⋮			
	Hist 1 Channel 1 bin $H1_1$		`UINT16`	2
	HIst 1 Channel 2 bin 1		`UINT16`	2
	⋮			
	Hist 1 Channel 2 bin $H1_2$		`UINT16`	2
	Hist 1 Channel 3 bin 1		`UINT16`	2
	⋮			
	Hist 1 Channel 3 bin $H1_3$		`UINT16`	2
	⋮			
	⋮			
	Colour space N	enumer.	`UINT16`	2
	Hist N No. of channel 1 bins (HN_1)		`UINT16`	2
	Hist N No. of channel 2 bins (HN_2)		`UINT16`	2
	Hist N No. of channel 3 bins (HN_3)		`UINT16`	2
	Hist N Channel 1 bin 1		`UINT16`	2
	⋮			
	Hist N Channel 1 bin HN_1		`UINT16`	2
	HIst N Channel 2 bin 1		`UINT16`	2
	⋮			
	Hist N Channel 2 bin HN_2		`UINT16`	2
	Hist N Channel 3 bin 1		`UINT16`	2
	⋮			
	Hist N Channel 3 bin HN_3		`UINT16`	2

If only a monochrome histogram is required, use the first data channel. The list of supported colour spaces can be found in appendix D.

7.9.31 Trajectory of image template fields (template)

If bits T0 and T9 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Rep.	Type	Size (bytes)
H3 & T0 & T9	Template 1 width u_1 (pixels)		`UINT16`	2
	Template 1 height v_1 (pixels)		`UINT16`	2
	Template 1 pixel packing	enumer.	`UINT16`	2
	Template 1 data		`UCHAR`	$u_1 \times v_1$
	⋮			
	Template N width u_N (pixels)		`UINT16`	2
	Template N height v_N (pixels)		`UINT16`	2
	Template N pixel packing	enumer.	`UINT16`	2
	Template N data		`UCHAR`	$u_N \times v_N$

Currently supported pixel packing values can be found in appendix D.

7.9.32 Trajectory of depth map fields (depth)

If bits T0 and T10 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Rep.	Type	Size (bytes)
H3 & T0 & T10	Depth map 1 width u_1 (pixels)		`UINT16`	2
	Depth map 1 height v_1 (pixels)		`UINT16`	2
	Depth map 1 pixel packing	enumer.	`UINT16`	2
	Depth map 1 data		`UCHAR`	$u_1 \times v_1$
	⋮			
	Depth map N width u_N (pixels)		`UINT16`	2
	Depth map N height v_N (pixels)		`UINT16`	2
	Depth map N pixel packing	enumer.	`UINT16`	2
	Depth map N data		`UCHAR`	$u_N \times v_N$

Currently supported pixel packing values can be found in appendix D.

7.9.33 Trajectory of behaviour fields (behav)

If bits T0 and T11 of the target trajectory bitmask are set, these fields will be included in the target data block

Bitmask	Field	Representation	Type	Size (bytes)
H3 & T0 & T11	Behaviour 1	enumeration	`UINT16`	2
	⋮			
	Behaviour N	enumeration	`UINT16`	2

Currently supported behaviour values can be found in appendix D.

7.10 The event section

This section contains any information relating to specific events.

Bitmask	Field	Representation	Type	Size (bytes)
H4	Number of Event data blocks		`UINT16`	2
	EVENT 1 DATA BLOCK			
H4	Number of bytes		`UINT16`	2
	Event ID		`UINT16`	2
	Event type	enumeration	`UINT16`	2
	Length of text label (N)		`UINT16`	2
	Text label		`UCHAR`	N
	E - Event data bitmask	bitmask	`UINT32`	4
H4 & E0	**Optional** Position of event in image coords fields			
H4 & E1	**Optional** Position of event in LVCS coords fields			
H4 & E2	**Optional** List of targets involved in event fields			
	⋮			
	EVENT N DATA BLOCK			

7.10.1 Number of event data blocks

This field denotes the number of event data blocks which are present in the event section. This number and all subsequent event data will only be present in the packet if bit H4 is set in the header bitmask.

7.10.2 Number of bytes in an event block

This field contains the number of bytes in its own event data block, **NOT** the total event section.

7.10.3 Event ID

Each VSAM entity will keep a count of the event data they issue and this number will be used as a unique event identifier.

7.10.4 Imagery type

A list of currently supported events can be found in appendix E.

7.10.5 E - Event data bitmask

This bitmask describes which data fields are physically present in an event data block.

Bit	E31	E30	E29	E28	E27	E26	E25	E24
Mask	—	—	—	—	—	—	—	—
Bit	E23	E22	E21	E20	E19	E18	E17	E16
Mask	—	—	—	—	—	—	—	—
Bit	E15	E14	E13	E12	E11	E10	E9	E8
Mask	—	—	—	—	—	—	—	—
Bit	E7	E6	E5	E4	E3	E2	E1	E0
Mask	—	—	—	—	—	targets	lvcs pos	img pos

7.10.6 Event position in image coords fields (img pos)

If the E0 bit of the event data bitmask is set, these fields will be included in the event data block

Bitmask	Field	Representation	Type	Size (bytes)
H4 & E0	Event position u (pixels)		`UINT16`	2
	Event position v (pixels)		`UINT16`	2
	Event range r (metres)		`FLOAT32`	4
	Event time t	timestamp	`UINT32`	4

7.10.7 Event position in LVCS coords fields (lvcs pos)

If the E1 bit of the event data bitmask is set, these fields will be included in the event data block

Bitmask	Field	Representation	Type	Size (bytes)
H4 & E1	Event position x (metres)		`FLOAT32`	4
	Event position y (metres)		`FLOAT32`	4
	Event position z (metres)		`FLOAT32`	4
	Event time t	timestamp	`UINT32`	4

7.10.8 Target list involved in event fields (targets)

If the E2 bit of the event data bitmask is set, these fields will be included in the event data block

Bitmask	Field	Representation	Type	Size (bytes)
H4 & E2	Number of targets involved (N)		`UINT16`	2
	Target ID 1		`UINT16`	2
	⋮			
	Target ID N		`UINT16`	2

8 VSAM DIS packet structures

In this implementation of the VSAM network, communication between operator control units (OCUs) and visualisation nodes (VIS) is done using the distributed simulation (DIS) protocol on the UDP internet protocol. The DIS protocol is described in detail in the DIS protocol specification document, but presented here is a description of how information will be encoded in the DIS protocol to be sent from an OCU to a VIS node.

Field	Content	Representation	Type	Size (bytes)
Packet header	Protocol version (= 4)	enumeration	`UCHAR`	1
	Exercise ID (= 1)		`UCHAR`	1
	PDU type (= 20)		`UCHAR`	1
	Protocol family (= 5)		`UCHAR`	1
	Timestamp	timestamp	`UINT32`	4
	Packet length		`UINT16`	2
	Padding		`UINT16`	2
Source ID	Site		`UINT16`	2
	Source ID	enumeration	`UINT16`	2
	Entity		`UINT16`	2
Dest ID	Site		`UINT16`	2
	Dest ID	enumeration	`UINT16`	2
	Entity		`UINT16`	2
Request ID			`UINT32`	4
Padding			`UINT32`	4
Number of fixed datum records (N)			`UINT32`	4
Number of variable datum records (M)			`UINT32`	4
Fixed datum # 1	Datum ID	enumeration	`UINT32`	4
	Datum value		`INT32`	4
⋮				
Fixed datum # N	Datum ID	enumeration	`UINT32`	4
	Datum value		`INT32`	4
Variable datum # 1	Datum ID	enumeration	`UINT32`	4
	Datum length		`UINT32`	4
	Datum value		??	??
⋮				
Variable datum # N	Datum ID	enumeration	`UINT32`	4
	Datum length		`UINT32`	4
	Datum value		??	??

- Note 1: Timestamps are defined differently in the DIS protocol. A time represents the elapsed time since the beginning of the current hour, and the scale is such that $2^{31} - 1$ units = 1 hour (1 unit $\approx 1.68\mu$S).
- Note 2: Site and Entity values are TBD.
- Note 3: All fixed length data must be packed before variable length data.
- Note 4: It is not possible to pack multiple command, sensor, target, or event "sections" into a DIS packet. Therefore, a single packet is limited to a maximum of one command, sensor, target and/or event section.
- Note 5: Source and destination IDs can be found in appendix F.
- Note 6: All variable length data values must be multiples of 64 bits (8 bytes). If this does not naturally occur, padding must be added.

8.1 VSAM DIS fixed length data

Presented here are a list of fixed length data records which will be supported by the VSAM DIS packet structure. For a list of fixed length datum IDs, see appendix G.

8.1.1 Command ID and type

Field	Content	Representation	Type	Size (bytes)
Datum ID	DIS_FX_COMMAND_ID	enumeration	`UINT32`	4
Datum value	Command ID		`UINT16`	2
	Command type	enumeration	`UINT16`	2

Command IDs are generated by the originating entity. Command types can be found in appendix A.

8.1.2 Sensor ID and type

Field	Content	Representation	Type	Size (bytes)
Datum ID	DIS_FX_SENSOR_ID	enumeration	`UINT32`	4
Datum value	Sensor ID		`UINT16`	2
	Sensor type	enumeration	`UINT16`	2

Sensor IDs are unique identifiers for individual sensors. Sensor types can be found in appendix B.

8.1.3 Target ID and type

Field	Content	Representation	Type	Size (bytes)
Datum ID	DIS_FX_TARGET_ID	enumeration	`UINT32`	4
Datum value	Target ID		`UINT16`	2
	Target type	enumeration	`UINT16`	2

Target IDs are unique identifiers for targets and are maintained by the OCU. Target types can be found in appendix D.

8.1.4 Event ID and type

Field	Content	Representation	Type	Size (bytes)
Datum ID	DIS_FX_EVENT_ID	enumeration	`UINT32`	4
Datum value	Event ID		`UINT16`	2
	Event type	enumeration	`UINT16`	2

Event IDs are generated by the originating entity. Event types can be found in appendix E.

8.1.5 Target behaviour

Field	Content	Representation	Type	Size (bytes)
Datum ID	DIS_FX_TARGET_BEHAV	enumeration	`UINT32`	4
Datum value	Target behaviour	enumeration	`UINT16`	2
	padding		`UINT16`	2

A list of target behaviours can be found in appendix D.

8.2 VSAM DIS variable length data

Presented here are a list of variable length data records which will be supported by the VSAM DIS packet structure. For a list of variable length datum IDs, see appendix H.

8.2.1 Command target ID list

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_COMMAND_TARGET	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	No of targets		`UINT16`	2
	Target 1 ID		`UINT16`	2
	⋮			
	Target N ID		`UINT16`	2
	Padding			

Target IDs are unique identifiers for targets which are maintained by the OCU.

8.2.2 Command event ID list

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_COMMAND_EVENT	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	No of events		`UINT16`	2
	Event 1 ID		`UINT16`	2
	⋮			
	Event N ID		`UINT16`	2
	Padding			

Event IDs are unique to events and are maintained by the originating OCU.

8.2.3 Sensor text label

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_SENSOR_TEXT	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Text label		`UCHAR`	??
	Padding			

8.2.4 Sensor position and pose

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_SENSOR_POS	enumer.	`UINT32`	4
Datum Length	24 bytes		`UINT32`	4
Datum value	LVCS x (metres)		`FLOAT32`	4
	LVCS y (metres)		`FLOAT32`	4
	LVCS z (metres)		`FLOAT32`	4
	θ	Angle	`INT32`	4
	ϕ	Angle	`INT32`	4
	ρ	Angle	`INT32`	4

8.2.5 Sensor velocity and angular velocity

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_SENSOR_VEL	enumer.	UINT32	4
Datum Length	24 bytes		UINT32	4
Datum value	LVCS $\dot{x}$ (metres/sec)		FLOAT32	4
	LVCS $\dot{y}$ (metres/sec)		FLOAT32	4
	LVCS $\dot{z}$ (metres/sec)		FLOAT32	4
	$\dot{\theta}$	Angular velocity	FLOAT32	4
	$\dot{\phi}$	Angular velocity	FLOAT32	4
	$\dot{\rho}$	Angular velocity	FLOAT32	4

8.2.6 Sensor field of view

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_SENSOR_FOV	enumer.	UINT32	4
Datum Length	8 bytes		UINT32	4
Datum value	Horizontal H-fov	Angle	INT32	4
	Vertical V-fov	Angle	INT32	4

8.2.7 Sensor region of interest

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_SENSOR_ROI	enumer.	UINT32	4
Datum Length	??		UINT32	4
Datum value	Number of LVCS points		UINT16	2
	Point 1 x (metres)		FLOAT32	4
	Point 1 y (metres)		FLOAT32	4
	Point 1 z (metres)		FLOAT32	4
	⋮			
	Point N x (metres)		FLOAT32	4
	Point N y (metres)		FLOAT32	4
	Point N z (metres)		FLOAT32	4
	padding			

8.2.8 Sensor footprint

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_SENSOR_FP	enumer.	UINT32	4
Datum Length	??		UINT32	4
Datum value	Number of LVCS points		UINT16	2
	Point 1 x (metres)		FLOAT32	4
	Point 1 y (metres)		FLOAT32	4
	Point 1 z (metres)		FLOAT32	4
	⋮			
	Point N x (metres)		FLOAT32	4
	Point N y (metres)		FLOAT32	4
	Point N z (metres)		FLOAT32	4
	padding			

8.2.9 Sensor limits

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_SENSOR_LIMIT	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Number of LVCS points		`UINT16`	2
	Point 1 x (metres)		`FLOAT32`	4
	Point 1 y (metres)		`FLOAT32`	4
	Point 1 z (metres)		`FLOAT32`	4
	⋮			
	Point N x (metres)		`FLOAT32`	4
	Point N y (metres)		`FLOAT32`	4
	Point N z (metres)		`FLOAT32`	4
	padding			

8.2.10 Target text label

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_TARGET_TEXT	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Text label		`UCHAR`	??
	padding			

8.2.11 Target position

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_TARGET_POS	enumer.	`UINT32`	4
Datum Length	16 bytes		`UINT32`	4
Datum value	LVCS x (metres)		`FLOAT32`	4
	LVCS y (metres)		`FLOAT32`	4
	LVCS z (metres)		`FLOAT32`	4
	padding			4

8.2.12 Target velocity

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_TARGET_VEL	enumer.	`UINT32`	4
Datum Length	16 bytes		`UINT32`	4
Datum value	LVCS $\dot{x}$ (metres)		`FLOAT32`	4
	LVCS $\dot{y}$ (metres)		`FLOAT32`	4
	LVCS $\dot{z}$ (metres)		`FLOAT32`	4
	padding			4

8.2.13 Target image template

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_TARGET_IMG	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Image width u (pixels)		`UINT16`	2
	Image height v (pixels)		`UINT16`	2
	Pixel packing	enumer.	`UINT16`	2
	Data		`UCHAR`	N
	padding			

Image pixel packing types are listed in appendix D.

8.2.14 Target trajectory timestamps

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_TARGET_TRAJ	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Length of trajectory		`UINT16`	2
	Time 1	timestamp	`UINT32`	4
	⋮			
	Time N	timestamp	`UINT32`	4
	padding			

8.2.15 Target trajectory of positions

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_TARGET_TPOS	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	LVCS 1 x (metres)		`FLOAT32`	4
	LVCS 1 y (metres)		`FLOAT32`	4
	LVCS 1 z (metres)		`FLOAT32`	4
	⋮			
	LVCS N x (metres)		`FLOAT32`	4
	LVCS N y (metres)		`FLOAT32`	4
	LVCS N z (metres)		`FLOAT32`	4
	padding			4

8.2.16 Target trajectory of image templates

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_TARGET_TIMG	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Image 1 width u (pixels)		`UINT16`	2
	Image 1 height v (pixels)		`UINT16`	2
	Image 1 pixel packing	enumer.	`UINT16`	2
	Image 1 data		`UCHAR`	$u \times v$
	⋮			
	Image N width u (pixels)		`UINT16`	2
	Image N height v (pixels)		`UINT16`	2
	Image N pixel packing	enumer.	`UINT16`	2
	Image N data		`UCHAR`	$u \times v$
	padding			4

A list of currently supported image pixel packing types can be found in appendix D.

8.2.17 Target trajectory of behaviours

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_TARGET_TBEHAV	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Behaviour 1	enumer.	`UINT16`	2
	⋮			
	Behaviour N	enumer.	`UINT16`	2
	padding			4

A list of currently supported target behaviour types can be found in appendix D.

8.2.18 Event text label

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_EVENT_TEXT	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Text label padding		`UCHAR`	??

8.2.19 Event position/time

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_EVENT_POS	enumer.	`UINT32`	4
Datum Length	16		`UINT32`	4
Datum value	Event LVCS x (metres)		`FLOAT32`	4
	Event LVCS y (metres)		`FLOAT32`	4
	Event LVCS z (metres)		`FLOAT32`	4
	Event time t	timestamp	`FLOAT32`	4

8.2.20 Event target list

Field	Content	Rep.	Type	Size (bytes)
Datum ID	DIS_VA_EVENT_TARGET	enumer.	`UINT32`	4
Datum Length	??		`UINT32`	4
Datum value	Event number of targets		`UINT16`	2
	Target 1 ID		`UINT16`	2
	$\vdots$			
	Target 1 ID		`UINT16`	2
	padding			

Target IDs are unique identifiers maintained by the OCU.

A Currently supported command data

Enumeration	Description
CM_I_AM_ALIVE	Report availability of sensor
CM_SET_BANDWIDTH	Limit bandwidth used by sensor
CM_SHUT_UP	Stop sensor from transmitting data
CM_GENERAL_ACK	Acknowledge command
CM_NET_ERROR	Report error has occured

Enumeration	Description
CM_REQUEST_DATA	Request data from destination node
CM_REQUEST_DATA_PRIORITY	Urgently request data from destination node
CM_SET_DATA	Set values of targets/sensors, etc.
CM_SET_PACKET_DESCRIPTION	Set capability of sensor

Enumeration	Description
CM_NO_OPERATION	Null command
CM_NO_RESPONSE	Report inability to complete command
CM_PARTIAL_RESPONSE	Report part of command was completed
CM_FULL_RESPONSE	Report all of command was completed
CM_SET_IDLE_POS	Set idling position for sensor
CM_RESET_IDLE_POS	Reset idling position for sensor
CM_SET_IDLE_BEHAV	Set idling behaviour for sensor
CM_RESET_IDLE_BEHAV	Reset idling behaviour for sensor
CM_SET_HANDOFF	Set regions for handoff
CM_RESET_HANDOFF	Reset handoff regions
CM_REQUEST_HANDOFF	Request handoff between sensors
CM_NEW_TARGET	New target identified by SPU
CM_TRACK_TARGET	Track specific target or targets
CM_STOP_TRACK_TARGET	Stop tracking current target
CM_SEARCH_TARGET	Search for specific target or targets
CM_STOP_SEARCH_TARGET	Stop searching for specific target or targets
CM_SEARCH_EVENT	Search for specific event
CM_STOP_SEARCH_EVENT	Stop searching for specific event

B Currently supported sensor data

Sensor type	Description
TY_UNCLASSIFIED	Unknown sensor type
TY_RGB_VIDEO	Regular color video camera
TY_MONO_VIDEO	Regular monochrome video camera
TY_IR_VIDEO	Near IR video camera
TY_THERMAL_VIDEO	Far IR/thermal video camera
TY_IFD_SPU_TYPE_I	IFD SPU v1.0 1997
TY_IFD_SPU_TYPE_II	IFD SPU v2.0 1998
TY_IFD_SPU_TYPE_III	IFD Rapid deployment SPU
TY_IFD_SPU_AIRBORNE	IFD Airborne SPU
TY_TI_SPU	Texas Instruments SPU
TY_OMNI_SPU	Lehigh/Columbia Paracam SPU
TY_SVM_SPU	SRI Small vision module SPU

Sensor ID	Type	Location
ID_SPU_1	TY_IFD_SPU_TYPE_II	Planetary robotics building
ID_SPU_2	TY_IFD_SPU_TYPE_II	Smith Hall
ID_SPU_3	TY_IFD_SPU_TYPE_II	Wean Hall
ID_SPU_4	TY_IFD_SPU_TYPE_II	Roberts Hall
ID_SPU_5	TY_OMNI_SPU	Roberts Hall walkway
ID_SPU_6	TY_IFD_SPU_TYPE_II	Porter Hall
ID_SPU_7	TY_IFD_SPU_TYPE_III	Relocatable (Frew St)
ID_SPU_8	TY_IFD_SPU_AIRBORNE	Airborne (NVL Islander)
ID_SPU_9	TY_TI_SPU	Planetary robotics building
ID_SPU_10	TY_SVM_SPU	TBD

Sensor status	Description
STAT_STANDBY	Standing by, not yet functional
STAT_IDLE	Sensor idle
STAT_TRACKING	Tracking a target
STAT_MULTI_TRACKING	Tracking multiple targets
STAT_SEARCHING	Waiting for specific targets or events

C Currently supported imagery data

Still image type	Description
IMG_RAW_MONO	Raw monochrome image 8-bit pixels
IMG_RAW_RGB	Raw rgb image 24-bit pixels
IMG_RAW_RGBA	Raw rgba image 32-bit pixels
IMG_RAW_ABGR	Raw abgr image 32-bit pixels
IMG_GIF	GIF format image
IMG_PGM	PGM image
IMG_JPEG	JPEG family compressed image

Video clip type	Description
MV_AVI	AVI movie
MV_QT	Quick time
MV_MJPEG	MJPEG family of movie compression
MV_MPEG	MPEG family of movie compression
MV_SGI	SGI movie format

D Currently supported target data

Target type	Description
TG_UNCLASSIFIED	Unknown target type
TG_HUMAN	Human target, intent unknown
TG_HUMAN_BLUE	Friendly human
TG_HUMAN_RED	Unfriendly human
TG_HUMAN_GROUP	Human group target, intent unknown
TG_HUMAN_GROUP_BLUE	Friendly human group
TG_HUMAN_GROUP_RED	Unfriendly human group
TG_VEHICLE	Vehicle target, type/intent unknown
TG_VEHICLE_BLUE	Friendly vehicle target
TG_VEHICLE_RED	Unfriendly vehicle target
TG_CAR	Car target, type/intent unknown
TG_CAR_BLUE	Friendly car target
TG_CAR_RED	Unfriendly car target
TG_TRUCK	Truck target, type/intent unknown
TG_TRUCK_BLUE	Friendly truck target
TG_TRUCK_RED	Unfriendly truck target

Colour space	Description
COL_MONOCHROME	Monochrome histogram (use first channel for packing)
COL_RGB	RGB colour space
COL_RGB_NORMALISED	Normalised RGB colour space (r = R/(R+G+B))
COL_YUV	YUV colour space
COL_HSV	HSV colour space

Pixel packing	Description
PIX_MONOCHROME	8-bit pixels monochrome
PIX_RGB	24 bit pixels RGB
PIX_RGBA	32 bit pixels RGBA
PIX_ABGR	32 bit pixels ABGR (SGI standard)

Target behaviour	Description
BH_UNKNOWN	Unknown target behaviour
BH_HUMAN_STANDING	Immobile human
BH_HUMAN_WALKING	Human walking
BH_HUMAN_RUNNING	Human running
BH_VEHICLE_PARKED	Vehicle parked
BH_VEHICLE_MOTION	Vehicle in motion

E Currently supported event data

Event type	Description
EV_UNCLASSIFIED	Unknown event
EV_RENDEZVOUS	Two human targets interact
EV_HVI_ENTRY	Human enters vehicle
EV_HVI_EXIT	Human leaves vehicle
EV_LOITER	Human loiters in specified area
EV_CAR_ALARM	Attempted car theft
EV_CONFLICT	interaction between red and blue forces

F Currently supported entity IDs

Entity ID	Description
ID_OCU_1	VSAM IFD OCU
ID_GUI_1	VSAM IFD GUI
ID_VIS_1	VSAM IFD VIS node network
ID_VIS_SEEDS	SEEDS consortium VIS node
ID_SPU_1	Planetary robotics building SPU
ID_SPU_2	Smith Hall SPU
ID_SPU_3	Wean Hall SPU
ID_SPU_4	Roberts Hall SPU
ID_SPU_5	Roberts Hall walkway Lehigh/Columbia FRE-SPU
ID_SPU_6	Porter Hall SPU
ID_SPU_7	Relocatable (Frew St) SPU
ID_SPU_8	Airborne (NVL Islander) SPU
ID_SPU_9	Planetary robotics building TI FRE-SPU
ID_SPU_10	SRI FRE-SPU

G Currently supported DIS fixed datum IDs

Fixed datum ID	Description
DIS_FX_COMMAND_ID	Command ID and type
DIS_FX_SENSOR_ID	Sensor ID and type
DIS_FX_TARGET_ID	Target ID and type
DIS_FX_TARGET_BEHAV	Target behaviour
DIS_FX_EVENT_ID	Event ID and type

H Currently supported DIS variable datum IDs

Fixed datum ID	Description
DIS_VA_COMMAND_TARGET	Command ID and type
DIS_VA_COMMAND_EVENT	Sensor ID and type
DIS_VA_SENSOR_TEXT	Sensor text label
DIS_VA_SENSOR_POS	Sensor position in LVCS
DIS_VA_SENSOR_VEL	Sensor velocity in LVCS
DIS_VA_SENSOR_FOV	Sensor field of view
DIS_VA_SENSOR_ROI	Sensor region of interest in LVCS
DIS_VA_SENSOR_FP	Sensor footprint in LVCS
DIS_VA_SENSOR_LIMIT	Sensor limits in LVCS
DIS_VA_TARGET_TEXT	Target text label
DIS_VA_TARGET_POS	Target position in LVCS
DIS_VA_TARGET_VEL	Target velocity in LVCS
DIS_VA_TARGET_IMG	Target image template
DIS_VA_TARGET_TRAJ	Target trajectory length/timestamp
DIS_VA_TARGET_TPOS	Target trajectory of positions
DIS_VA_TARGET_TIMG	Target trajectory of image templates
DIS_VA_TARGET_TBEHAV	Target trajectory of behaviour
DIS_VA_EVENT_TEXT	Event text label
DIS_VA_EVENT_POS	Event position in LVCS
DIS_VA_EVENT_TARGET	Event list of relevan targets