MicroPilot VSM User Guide

UgCS 3.3.362

Contents

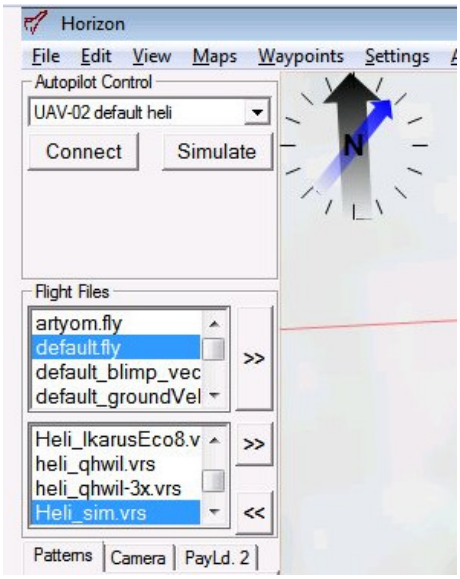
1 MicroPilot VSM User Guide

1.1 First time vehicle connection

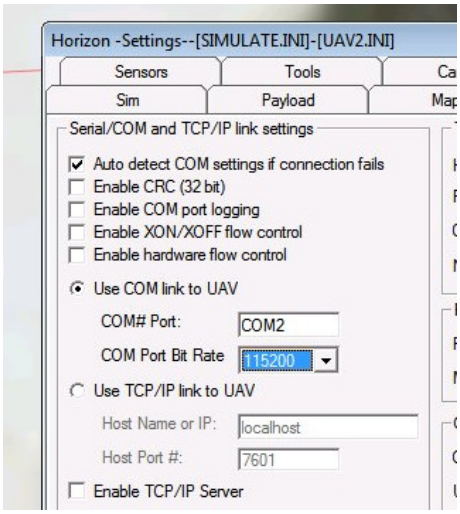
See Disclaimer.

The following instructions are applicable for Horizon simulator only. For working with real Micropilot hardware they may significantly differ due to high flexibility of the autopilot. Please contact UgCS support for assistance.

Select these built-in profiles in Horizon:



Set up serial port for simulation in Horizon settings "Comms" tab:

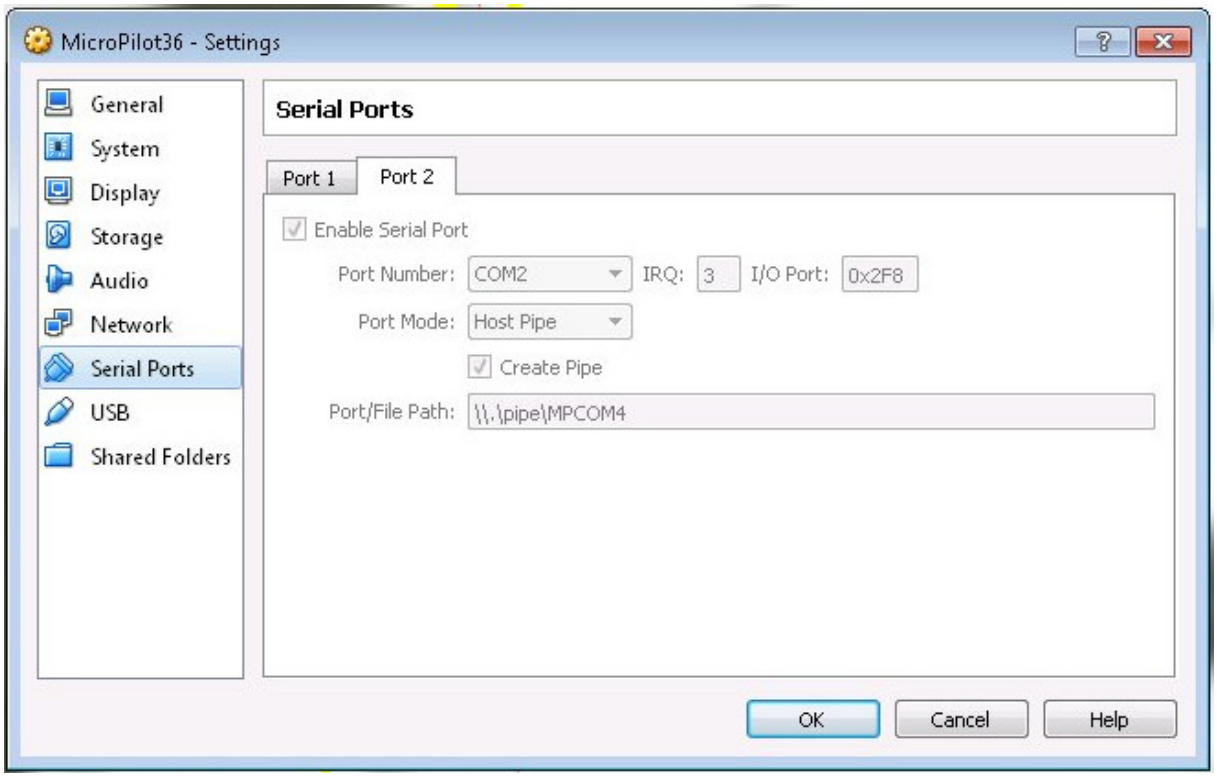


After that you can start simulation by pressing "Simulate" button. The simulated autopilot will be available on the specified serial port. You can connect to it by using some other serial port hardware connected to computer which runs Micropilot VSM. The VSM should have proper port settings which correspond to settings entered in Horizon. See Serial port configuration section for serial port configuration details in the VSM.

1.1.1 Setup example with virtual machine

Since Horizon simulator provides so ugly interface for accessing simulated vehicle (via hardware serial port), some technique can be used to make it more convenient. It might be using some software which creates virtual serial port (or virtually connected ports pair). Another way is to use virtual machine and its emulated serial port. VirtualBox is used in this example.

Configure emulated serial port. Map it to pipe.



VSM does not support working with pipes so some utility software should be used for proxying data between a pipe and TCP connection. "piper" utility was created for this task (based on VSM SDK) and is available for download on GitHub: https://github.com/UgCS/piper/releases. (Here is other alternative: http://www.hw-group.com/products/hw_vsp/index_en.html#download). Example of configuration ("piper.conf" file which is located aside with piper executable):

```
pipe.1.first.type = pipe
pipe.1.first.name = \\\\.\\pipe\\MPPORT4
pipe.1.second.type = tcp_in
pipe.1.second.local_port = 60004
```

This configuration instructs the utility to open pipe "\\\\.\\pipe\\MPPORT4" and proxy it to TCP port 60004 (on all interfaces). Then you can connect VSM to that port by placing the following lines in VSM configruation:

```
connection.tcp_out.1.address = 10.0.0.1
connection.tcp_out.1.port = 60004
```

IP-address should correspond to machine where the "piper" is running.

1.2 Mission execution specifics

Command	Support	Notes
ARM	Yes	After armed the vehicle starts counting down. If take-off has not been issued after 30 seconds the vehicle is disarmed automatically.

DISARM	No	
AUTOMODE	Partial	Can be used once right after arming to take-off and execute the mission. There is some time after the take-off when the vehicle does not accept any command.
MANUALMODE	No	
CLICK-TO-GO	Partial	Works but has some specifics. Can be issued during mission execution. Once issued, the vehicle should reach the specified point before the mission can be continued by CONTINUE command. Next "Click to go" can be issued only after the mission is continued. Heading specifying is not supported.
JOYSTICK	No	
HOLD	Yes	
CONTINUE	Yes	
RETURNHOME	Yes	
TAKEOFF	No	Send AUTOMODE to take-off and start the mission.
LAND	Yes	
EMERGENCYLAND	No	
CAMERA_TRIGGER	No	

Mission can be executed only once after powering on. When the mission is completed the vehicle lands, and a power cycle and a new mission uploading should be done to launch the vehicle again with a mission.

Flight plan element / action	Support	Notes
Camera control	Yes	
Camera trigger	No	
Panorama	No	
Take-off	Yes	Should be the first mission item if used.
Landing	Yes	Should be the last mission item if used.
Set camera by time	No	
Set camera by distance	No	
POI	Yes	
Change yaw	No	
Wait	Yes	

1.3 Fail-safe actions

GPS Lost:

Action	Result
Wait	Aircraft tries to maintain altitude
Land	Aircraft slowly decends

RC Lost:

Action	Result
Wait	Aircraft returns home and lands
Land	Aircraft returns home and lands

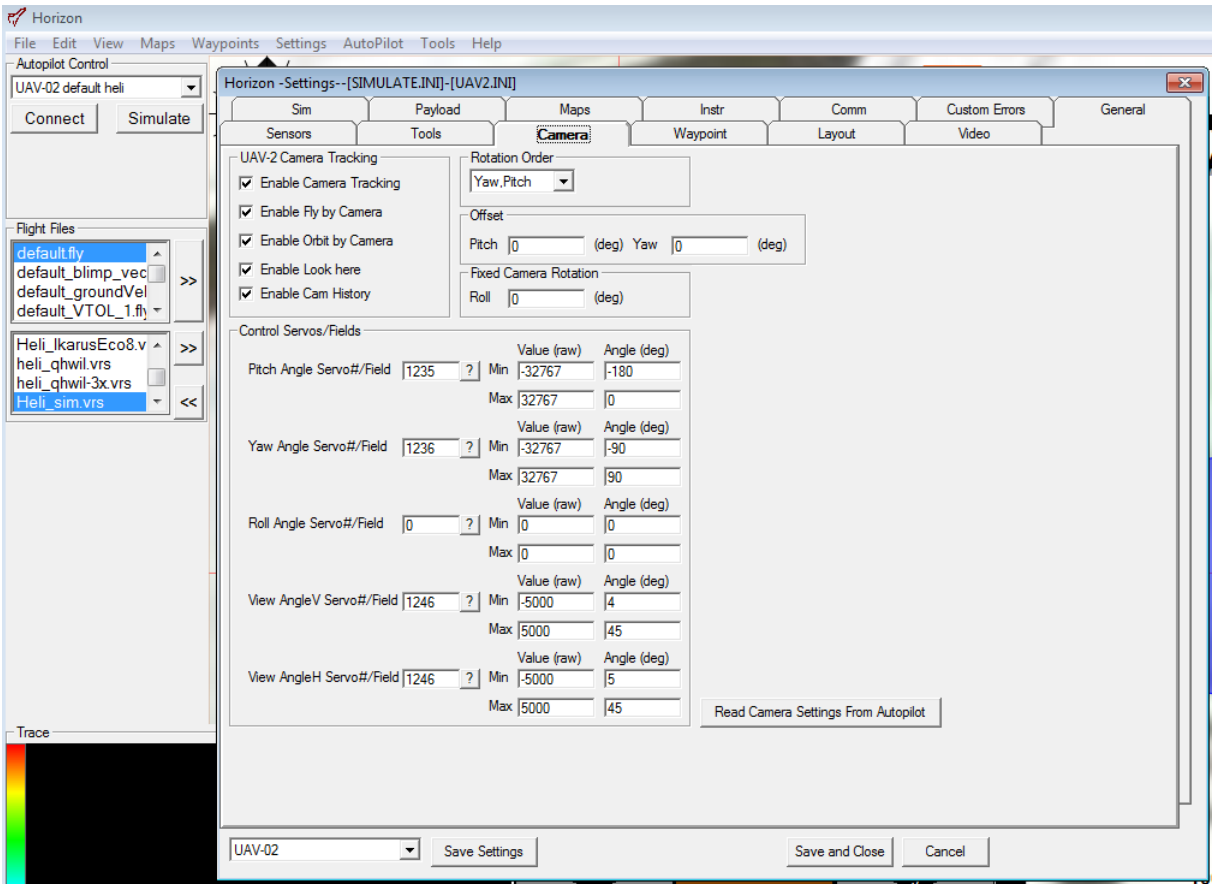
Return Home	Aircraft changes altitude to failsafe value and returns home
Continue	Aircaft continues mission

Battery Low:

Action	Result
Wait	Stop and wait for user interaction.
Land	Land at current position.
Return Home	If possible aircraft returns home and lands, if not possible slowly decends
Continue	Aircaft continues mission

1.4 Camera setup in Horizon when using simulator

In order to properly see camera footprint in Horizon simulation the camera servo should be configured as on the picture bellow:



Tested with Horizon 3.7.436.0.

1.5 Native route generation

VSM is able to generate route in Micropilot .fly format. It can be used to upload the route to the vechicle via Horizon application. Generated route covers only waypoints navigation and WP actions.

- Route is generated in metric units.
- Commands which can appear in generated fly file:

Command	Notes
metric	Fly file is generated in metric units.
[field] = value	Field Assignment Command. Used to set autopilot variables.
takeoff	Initiate vehicle take off.
climb	Initiate altitude change.
waitclimb	Climb to specified altitude before proceeding to next command.
wait	Wait for specified duration.
hoverAt	Start navigation to waypoint and hover at it.
circuit	Land
repeat -1	Defines the route end.

- Route features and corresponding generated commands

UgCS Feature	Generated commands	Notes
Takeoff waypoint	takeoff, climb, hoverAt	Take off and fly to the specified location.
Waypoint	climb, hoverAt	Fly to specified location and hover
Land waypoint	hoverAt,circuit	Fly to the specified location and land.
WP accetance radius	[wptDia] = <acceptance_radius ∗ 2>	Initiate altitude change.
WP flight speed	[cruiseSpd] = <speed>	Set speed.
Wait WP action	wait <duration>	Wait at waypoint.
POI WP action	[camTargetEnable] = 1 [camTargetEast] = <longitude> [camTargetNorth] = <latitude> [camTargetUp] = <altitude>	Initiate camera targeting and set absolute coordinates of POI.
Set camera attitude WP action	[fServo9], [fServo10]	Set camera angle using hardcoded fields which are configured for use with Horizon simulation.

- No failsafe options are generated.
- No threads or patterns are generated.

Example of generated route consiting of 4 waypoints and some WP actions in .fly format:

```
metric
[disableGcsCmds] = 255
takeoff
waitclimb 32.151859
[disableGcsCmds] = 0
[wptDia] = 2.000000
climb 32.151859
hoverAt (24.072405E,56.977959N)
[fServo9] = 32767
[fServo10] = 0
[cruiseSpd] = 17.998356
[wptDia] = 2.000000
climb 33.620640
hoverAt (24.073918E,56.977439N)
wait 10
[cruiseSpd] = 17.996920
[wptDia] = 2.000000
climb 34.678177
hoverAt (24.073254E,56.977075N)
[camTargetEnable] = 1
[camTargetEast] = 0.420174
[camTargetNorth] = 0.994436
[camTargetUp] = 22.817450
[cruiseSpd] = 17.975965
[wptDia] = 2.000000
```

```
climb 38.475319
hoverAt (24.072091E,56.977253N)
[camTargetEnable] = 0
repeat -1
```

1.6 Configuration file

Configuration file location:

- **On Microsoft Windows:**

```
C:\Program Files (x86)\UgCS\bin\vsm-micropilot.conf
```

- **On GNU/Linux:**

```
/etc/opt/ugcs/vsm-micropilot.conf
```

- **On Apple OS X:**

```
/Users/[user name]/Library/Application Support/UGCS/configuration/vsm-micropilot.conf
```

1.6.1 Common parameters

All VSMs share a common set of configuration file parameters described in Common configuration file parameters. MicroPilot VSM configuration file prefix is:

```
vehicle.micropilot
```

1.6.2 Protocol parameters

- **Name:** micropilot.use_crc
- **Description:** Use CRC-32 in autopilot messages when set to non-zero. Otherwise single byte checksum is used.
- **Example:**

```
micropilot.use_crc = 1
```

1.7 Common configuration file parameters

VSM configuration file is a text file specified via command line argument - *-config* of the VSM application. Example:

```
--config /etc/opt/ugcs/vsm-ardupilot.conf
```

Each configuration parameter is defined as a line in the configuration file with the following structure:

```
name1.name2....nameX = value
```

where name1, name2 ... nameX are arbitrary names separated by dots to divide a variable into logical blocks and a value which can be a number value or a text string depending on the context. See below the description about common VSM configuration parameters.

1.7.1 UgCS server configuration

VSM can connect to UgCS in two different ways:

- Listen for connection form the UgCS server. See Listening address and Listening port.

  When VSM is configured in listening mode automatic VSM discovery can be set up, too. See Automatic service discovery
- Initiate connection to UgCS server. See UgCS server address and UgCS server port.

At least one of the above must be configured for VSM to work.

1.7.1.1 Listening address

Optional.

- **Name:** ucs.local_listening_address = [IP address]
- **Description:** Local address to listen for incoming connections from UgCS server.
- **Default:** 0.0.0.0 (listen on all local addresses)
- **Example:** ucs.local_listening_address = 10.0.0.2

1.7.1.2 Listening port

Optional.

- **Name:** ucs.local_listening_port = [port number]
- **Description:** Local TCP port to listen for incoming connections from UgCS server.
- **Example:** ucs.local_listening_port = 5556

1.7.1.3 UgCS server address

Optional.

- **Name:** ucs.address = [IP address]
- **Description:** UgCS server address to connect to.
- **Example:** ucs.address = 1.2.3.4

1.7.1.4 UgCS server port

Optional.

- **Name:** ucs.port = [port number]
- **Description:** UgCS server port.
- **Example:** ucs.port = 3335

1.7.1.5 Retry timeout

Optional.

- **Name:** ucs.retry_timeout = [seconds]
- **Description:** Retry timeout for outgoing server connections in seconds.
- **Default:** 10
- **Example:** retry_timeout = 11

1.7.2 Automatic service discovery

VSM can respond to automatic service discovery requests form UgCS server.

When this parameter is not configured, service discovery is disabled.

Optional.

- **Name:** service_discovery.vsm_name = [Service name]
- **Description:** Human readable service name.
- **Example:** service_discovery.vsm_name = Ardupilot VSM

1.7.3 Logging configuration

1.7.3.1 Level

Optional.

- **Name:** log.level = [error|warning|info|debug]
- **Description:** Logging level.
- **Default:** info
- **Example:** log.level = debug

1.7.3.2 File path

Optional.

- **Name:** log.file_path = [path to a file]
- **Description:** Absolute or relative (to the current directory) path to a logging file. Logging is disabled if logging file is not defined. File should be writable. Backslash should be escaped with a backslash.
- **Example:** log.file = /var/opt/ugcs/log/vsm-ardupilot/vsm-ardupilot.log
- **Example:** log.file = C:\\Users\\John\\AppData\\Local\\UGCS\\logs\\vsm-ardupilot\\vsm-ardupilot.log

1.7.3.3 Maximum single file size

Optional.

- **Name:** log.single_max_size = [size]
- **Description:** Maximum size of a single log file. When maximum size is exceeded, existing file is renamed by adding a time stamp and logging is continued into the empty file. [size] should be defined as a number postfixed by a case insensitive multiplier:
  - Gb, G, Gbyte, Gbytes: for Giga-bytes
  - Mb, M, Mbyte, Mbytes: for Mega-bytes
  - Kb, K, Kbyte, Kbytes: for Kilo-bytes
  - no postfix: for bytes
- **Default:** 100 Mb
- **Example:** log.single_max_size = 500 Mb

1.7.3.4 Maximum number of old log files

Optional.

- **Name:** log.max_file_count = [number]
- **Description:** Log rotation feature. Maximum number of old log files to keep. After reaching single_max_size of current log file VSM will rename it with current time in extension and start new one. VSM will delete older logs so the number of old logs does not exceed the max_file_count.
- **Default:** 1
- **Example:** log.max_file_count = 5

1.7.4 Mission dump path

Optional.

- **Name:** [prefix].mission_dump_path = [path to a file]
- **Description:** File to dump all generated missions to. Timestamp is appended to the name. Delete the entry to disable mission dumping. All directories in the path to a file should be already created.
- **Example:** vehicle.ardupilot.mission_dump_path = C:\\tmp\\ardupilot_dump

1.7.5 Command execution control

When vehicle is connected via unreliable link VSM will retry each command several times before failing. This section describes the parameters which control the command execution.

1.7.5.1 Command try count

- **Name:** vechicle.command_try_count = <number of="" times>=""></number>
- **Description:** Number of times the command will be issued before declaring it as failed. Must be greater than zero.
- **Default:** 3
- **Example:** vechicle.command_try_count = 5

1.7.5.2 Command timeout

- **Name:** vechicle.command_timeout = <timeout in="" seconds>=""></timeout>
- **Description:** Time to wait for response on issued command before retrying.
- **Unit:** second
- **Default:** 1

1.7.5.3 Detection timeout

- **Name:** vehicle.detection_timeout = <timeout in="" seconds>=""></timeout>
- **Description:** Time to wait for vehicle to respond after connection is established.
- **Unit:** second
- **Default:** 6

1.7.5.4 Vehicle serial prefix

- **Name:** vehicle.serial_prefix = string
- **Description:** String value used as prefix for all vehicle serial numbers conected to this VSM. Can be used to connect vehicles with equal serial numbers to the same server via different VSMs
- **Default:** not defined
- **Example:** vehicle.serial_prefix = group1:

1.8 Communication with devices

VSM can communicate with Vehicle over different communication channels

Currently supported channels:

- Serial port, see Serial port configuration for details.
- Outgoing TCP, see Outgoing TCP connection configuration for details.
- Incoming TCP, see Incoming TCP connection configuration for details.
- Outgoing UDP, see Outgoing UDP connection configuration for details.
- Incoming UDP, see Incoming UDP connection configuration for details.
- Incoming UDP (any peer), see Incoming UDP connection configuration (any peer) for details.
- Named pipe , see Named pipes for details.
- vsm-proxy (XBee), see Proxy configuration for details.

1.8.1 Serial port configuration

VSM which communicates with vehicles via serial ports should define at least one serial port, otherwise VSM will not try to connect to the vehicles. Port name and baud rate should be both defined.

1.8.1.1 Port name

Required.

- **Name:** connection.serial.[index].name = [regular expression]
- **Description:** Ports which should be used to connect to the vehicles by given VSM. Port names are defined by a [regular expression] which can be used to define just a single port or create a port filtering regular expression. Expression is case insensitive on Windows. [index] is a arbitrary port indexing name.
- **Example:** connection.serial.1.name = /dev/ttyUSB[0-9]+|com[0-9]+
- **Example:** connection.serial.2.name = com42

1.8.1.2 Port baud rate

Required.

- **Name:** connection.serial.[index].baud.[baud index] = [baud]
- **Description:** Baud rate for port opening. [baud index] is an optional arbitrary name used when it is necessary to open the same serial port using multiple baud rates. [index] is an arbitrary port indexing name.
- **Example:** connection.serial.1.baud.1 = 9600
- **Example:** connection.serial.1.baud.2 = 57600
- **Example:** connection.serial.2.baud = 38400

1.8.1.3 Excluded port name

Optional.

- **Name:** connection.serial.exclude.[exclude index] = [regular expression]
- **Description:** Ports which should not be used for vehicle access by this VSM. Port names are defined by a [regular expression] which can be used to define just a single port or create a port filtering regular expression. Filter is case insensitive on Windows. [exclude index] is a arbitrary indexing name used when more than one exclude names are defined.
- **Example:** connection.serial.exclude.1 = /dev/ttyS.∗
- **Example:** connection.serial.exclude = com1

1.8.1.4 Serial port arbiter

Optional.

- **Name:** connection.serial.use_arbiter = [yes|no]
- **Description:** Enable (yes) or disable (no) serial port access arbitration between VSMs running on the same machine. It is recommended to have it enabled to avoid situation when multiple VSMs try to open the same port simultaneously.
- **Default:** yes
- **Example:** connection.serial.use_arbiter = no

1.8.2 Outgoing TCP connection configuration

VSM can be configured to connect to the vehicle via TCP. VSM will try to establish connection to the specified address:port.

Used to connect to vehicle simulator or when vehicle is equipped with WiFi adapter.

1.8.2.1 Remote TCP port

Required.

- **Name:** connection.tcp_out.[index].port = [port number]
- **Description:** Remote port to connect to.
- **Example:** connection.tcp_out.1.port = 5762

1.8.2.2 IP-address for outgoing TCP connection

Required.

- **Name:** connection.tcp_out.[index].address = [IP-address]
- **Description:** IP-address of vehicle to connect to.
- **Example:** connection.tcp_out.1.address = 10.0.0.111

1.8.3 Incoming TCP connection configuration

VSM can be configured to listen for incoming TCP connections from the vehicle. Multiple vehicles are supported on the same port.

Used to connect to vehicle equipped with WiFi adapter.

1.8.3.1 Local listening TCP port

Required.

- **Name:** connection.tcp_in.[index].local_port = [port number]
- **Description:** Remote port to connect to.
- **Example:** connection.tcp_in.1.local_port = 5762

1.8.3.2 Local IP address

Optional.

- **Name:** connection.tcp_in.[index].local_address = [IP-address]
- **Description:** Local ip address to bind to.
- **Default:** 0.0.0.0 (all interfaces)
- **Example:** connection.tcp_in.1.local_address = 127.0.0.1

1.8.4 Outgoing UDP connection configuration

VSM can be configured to connect to the vehicle via UDP. VSM will try to establish UDP connection to the specified address:port.

1.8.4.1 Remote IP-address for UDP

Required.

- **Name:** connection.udp_out.[index].address = [IP-address]
- **Description:** Remote IP-address to send outgoing UDP packets to.
- **Example:** connection.udp_out.1.address = 192.168.1.1

1.8.4.2 Remote UDP port

Required.

- **Name:** connection.udp_out.[index].port = [port number]
- **Description:** Remote UDP port to send outgoing packets to.
- **Example:** connection.udp_out.1.port = 14551

1.8.4.3 Local IP-address for UDP

Optional.

- **Name:** connection.udp_out.[index].local_address = [IP-address]
- **Description:** Local ip address to bind to.
- **Default:** 0.0.0.0 (bind to all interfaces)
- **Example:** connection.udp_out.1.local_address = 0.0.0.0

1.8.4.4 Local UDP port

Optional.

- **Name:** connection.udp_out.[index].local_port = [port number]
- **Description:** Local UDP port to listen for incoming packets on.
- **Default:** 0 (bind to random port)
- **Example:** connection.udp_out.1.local_port = 14550

1.8.5 Incoming UDP connection configuration

VSM can be configured to listen for UDP conncections from the vehicle. Vehicle must be actively sending heartbeat/telemetry on specified UDP port before it can be detected by VSM. VSM will automatically detect multiple vehicles on the same port. This is very useful for "drone swarm" setups as there is no need to specify connector for each vehicle and no need to know the IP address of each vehicle in advance.

1.8.5.1 Local UDP port

Required.

- **Name:** connection.udp_in.[index].local_port = [port number]
- **Description:** Local UDP port to listen for incoming packets on.
- **Example:** connection.udp_in.1.local_port = 14550

1.8.5.2 Local IP-address for UDP

Optional.

- **Name:** connection.udp_in.[index].local_address = [IP-address]
- **Description:** Local ip address to bind to.
- **Default:** 0.0.0.0 (bind to all interfaces)
- **Example:** connection.udp_in.1.local_address = 0.0.0.0

1.8.6 Incoming UDP connection configuration (any peer)

This connection type is similar to "udp_in" with the exception that all incoming traffic will be recevied as one stream. It is used for special purpose connections and cannot be used to connect vehicles.

1.8.6.1 Local UDP port

Required.

- **Name:** connection.udp_any.[index].local_port = [port number]
- **Description:** Local UDP port to listen for incoming packets on.
- **Example:** connection.udp_any.1.local_port = 14550

1.8.6.2 Local IP-address for UDP

Optional.

- **Name:** connection.udp_any.[index].local_address = [IP-address]
- **Description:** Local ip address to bind to.
- **Default:** 0.0.0.0 (bind to all interfaces)
- **Example:** connection.udp_any.1.local_address = 0.0.0.0

1.8.7 Named pipes

VSM is able to communicate with vehicle over named pipe. The pipe must already exist.

- **Name:** connection.pipe.[index].port = pipe_name
- **Description:** Pipe name
- **Example:** connection.pipe.1.name = \\pipe\my_named_pipe

1.8.8 Proxy configuration

VSM is able to communicate with vehicle via proxy service which redirects dataflow received from vehicle through TCP connection to VSM and vice versa using specific protocol. In other words proxy service appears as a router between vehicle and VSM. At the moment there is one implementation of proxy in UgCS called XBee Connector which retranslates data from ZigBee network to respective VSM.

1.8.8.1 IP-address for proxy

Required.

- **Name:** connection.proxy.[index].address = [IP-address]
- **Description:** IP-address to connect proxy to. Specify local or remote address.
- **Example:** connection.proxy.1.address = 127.0.0.1

1.8.8.2 TCP port for proxy

Required.

- **Name:** connection.proxy.[index].port = [port number]
- **Description:** TCP port to be connected with proxy through. Should be the same as in congiguration on proxy side.
- **Example:** connection.proxy.1.port = 5566

2 Disclaimer

DISCLAIMER OF WARRANTIES AND LIMITATIONS ON LIABILITY.

(a) SMART PROJECTS HOLDINGS MAKE NO REPRESENTATIONS OR WARRANTIES REGARDING THE ACCURACY OR COMPLETENESS OF ANY CONTENT OR FUNCTIONALITY OF THE PRODUCT AND ITS DOCUMENTATION.

(b) SMART PROJECTS HOLDINGS DISCLAIM ALL WARRANTIES IN CONNECTION WITH THE PRODUCT, AND WILL NOT BE LIABLE FOR ANY DAMAGE OR LOSS RESULTING FROM YOUR USE OF THE PRODUCT. INCLUDING BUT NOT LIMITED TO INJURY OR DEATH OF USER OR ANY THIRD PERSONS OR DAMAGE TO PROPERTY.

(c) THE SOFTWARE IS SUPPLIED AS IS WITH NO WARRANTIES AND CAN BE USED ONLY AT USERS OWN RISK.