

УТВЕРЖДЕН

ЛУБА.00050-01 31 01-ЛУ

**КОМПЛЕКТ ПРОГРАММНЫЙ LORIS TOOL
ПРОГРАММНЫЙ ИНТЕРФЕЙС ПРИЛОЖЕНИЯ**

Описание применения

ЛУБА.00050-01 31 01

Листов 49

Литера «О₁»

Москва, 2026 г.

АННОТАЦИЯ

Настоящий документ разработан в соответствии с требованиями ГОСТ 19.101-2024, ГОСТ 19.106-78, ГОСТ 19.502-78 и распространяется на программный интерфейс приложения (Application Programming Interface, далее — API) комплекта программного LorisTool ЛУБА.00050-01 (далее — Программа).

Порядок установки, обновления, удаления и использования Программы оператором при эксплуатации изложен в Руководстве оператора ЛУБА.00050-01 34 01.

Предприятие-изготовитель: ООО «ЭЛИАРС».

Адрес предприятия-изготовителя: Российская Федерация, 124460, г. Москва, г. Зеленоград, ул. Конструктора Гуськова, дом 8, стр. 1.

Адрес для корреспонденции: 124460, г. Москва, г. Зеленоград, а/я 201.

Телефон (факс): +7 (499) 995-24-53.

Электронная почта: info@eliars.ru.

Техническая поддержка: support@eliars.ru.

Информация о продукте: <https://www.eliars.ru/lori>.

Текущая версия комплекта программного обеспечения, эксплуатационная документация, описание применения программного интерфейса приложения API, тестовые файлы данных: <https://www.loris-radar.ru>.

ИЗМЕНЕНИЯ

Дата	Версия документа	Изменения
26.06.2026	34	A.1 Подписка на монитор телеметрии Таблица A.1.4 «Структура данных TelemetryData»: добавлены поля со значениями внешних датчиков температуры электропривода <i>rotator_ext_temperature_1</i>, <i>rotator_ext_temperature_2</i>. A.2 Подписка на базовые параметры управления Таблица A.2.4 «Структура данных ParametersData», добавлены новые параметры: - координаты точки стояния РЛС (широта, долгота, азимут, высота); - запись сырых данных и траекторий; - информация об отключении излучения в секторах; - фазы приемных каналов; - сектор маскирования алгоритма обнаружения; - информация о когерентном интервале; - параметры управления порогами детектора, фильтров по скорости, дальности, высоте; - параметры кластеризации; - параметры классификатора; - параметры фильтра пассивных помех; - параметры автосопровождения; - параметры трансивера; - классы, отправляемые в API. A.3 Подписка на координаты РЛС Метод <i>LocationSubscription</i> (подписка на координаты РЛС) объявлен устаревшим и к использованию не рекомендуется. В новых разработках место него следует использовать соответствующие поля в методе <i>ParametersSubscription</i>. Поддержка метода <i>LocationSubscription</i> в API сохраняется. A.5 Команда установки значения параметра Таблица A.5.1 «Структура данных Request с методом SetParameter»: добавлены новые параметры управления. Перечень управляемых параметров соответствует приведенному в методе <i>ParametersSubscription</i> (таблица A.2.4). п. 4.3, таблица 1 Добавлены новые команды «Подписка на пресеты», «Команда изменения состояния пресета». Описание пресетов (предустановок алгоритма обработки) см. в Руководстве по эксплуатации. п. 4.4, таблица 2 Добавлено новое поле <i>presets</i>. Добавлен новый раздел A.7 Подписка на пресеты. Добавлен новый раздел A.8 Команда изменения состояния пресета
24.04.2026	35	A.1 Подписка на монитор телеметрии Таблица A.1.4 «Структура данных TelemetryData»: для полей <i>recv_speed</i>, <i>lost_frames_percent</i>, <i>lost_intervals</i> изменен тип данных из string в double

Содержание

1 Назначение Программы.....	5
2 Условия применения	5
3 Описание задачи.....	6
4 Входные и выходные данные.....	6
Приложение А (обязательное) Описание команд, ответов и структур данных	10
A.1 Подписка на монитор телеметрии	10
A.2 Подписка на базовые параметры управления	17
A.3 Подписка на координаты РЛС	23
A.4 Подписка на поток траекторий	25
A.5 Команда установки значения параметра	32
A.6 Команда отказа от подписки	34
A.7 Подписка на пресеты	36
A.8 Команда изменения состояния пресета.....	40
Приложение Б (справочное) Использование приложения LorisTool для просмотра имитированных траекторий.....	42
Б.1 Имитированные траектории	42
Б.2 Проигрывание траектории из файла	44
Б.3 Проигрывание файла сырых данных с помощью приложения Vacon player.....	45
Приложение В (справочное) Включение, проверка работоспособности и отключение РЛС с использованием API	46

1 Назначение Программы

1.1 Программа выполняется на вычислительном блоке (серверном приложении Loris Tool) и реализует API, предназначенный для подключения внешних потребителей (клиентских приложений) с целями:

1. получения телеметрических данных радиолокационной станции (РЛС);
2. получения основных параметров управления РЛС;
3. получения координат точки стояния РЛС;
4. получения потока координатных точек (траекторий) обнаруженных целей;
5. установки значений параметров управления РЛС.

1.2 Обобщенная функциональная схема взаимодействия Программы с РЛС и другими программными компонентами приведена на рисунке 1.

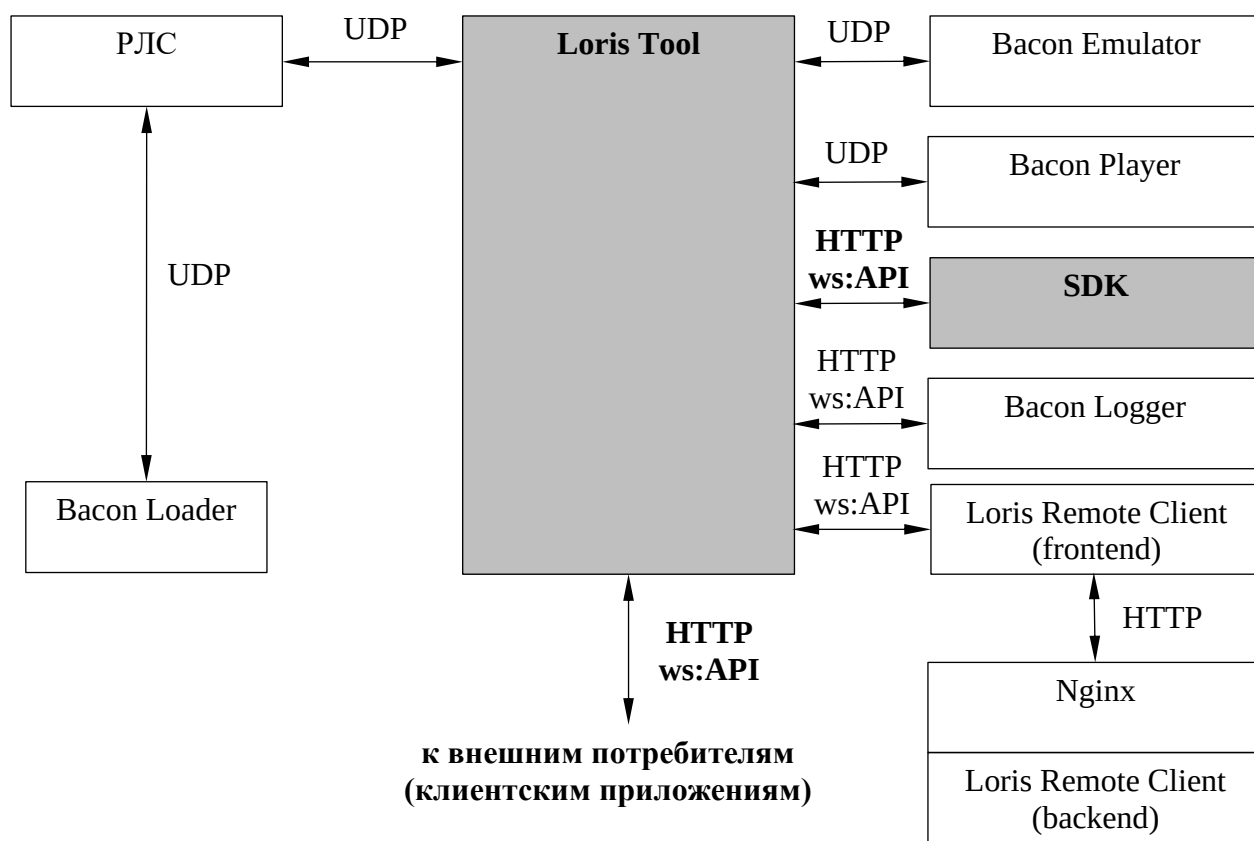


Рисунок 1 — Схема функциональная обобщенная взаимодействия Программы с РЛС и другими программными компонентами

2 Условия применения

2.1 API основан на потоковой передаче данных через WebSockets в формате JSON.

3 Описание задачи

3.1 Базовый принцип работы с API заключается в том, что клиентское приложение устанавливает соединение с сервером через точку входа: **ws://{loris-address}:7007/api/v1**.

Далее клиентское приложение посылает команды в установленное соединение для получения или изменения данных. Все обмены происходят в текстовом виде.

3.2 Принцип получения данных: для получения данных клиентскому приложению необходимо сформировать команду подписки на нужную группу данных и отправить команду в WebSocket. Далее Программа возвращает в WebSocket запрошенные данные. После установки соединения в первой посылке клиентское приложение получает полный набор запрошенных данных, в последующих посылках клиентское приложение получает только измененные данные.

3.3 Принцип изменения данных: для изменения данных клиентскому приложению необходимо сформировать команду с указанием группы изменяемых данных, а также перечня самих данных. Команда с данными отправляется в WebSocket. В ответ Программа отправляет подтверждение об изменении данных.

3.4 Клиентское приложение может отписаться от ранее сформированной подписки без закрытия соединения, послав специальную команду с номером подписки. Ранее установленное соединение может быть использовано как описано в 3.1. Если клиентское приложение закрывает соединение, то команда отказа от подписки не является обязательной.

4 Входные и выходные данные

4.1 Входными данными Программы являются команды, направляемые от клиентского приложения с использованием API:

- команды подписки на группу данных;
- команда изменения данных (установки значения параметра);
- команда отказа от подписки.

4.2 Выходными данными Программы являются ответы на команды клиентского приложения, отправляемые с использованием API:

- ответ на команды подписки;
- структуры запрошенных данных;
- структуры, описывающие подтверждение или отклонение изменения данных.

4.3 Перечень команд API (запрос, Request) в соответствии с таблицей 1; **"someid"** — уникальный идентификатор запроса. Описание команд, ответов и структур данных приведены в Приложении А.

Таблица 1 — Перечень команд клиентского приложения (Request)

Описание команды	Структура данных, передаваемая от клиентского приложения	Примечание
1 Подписка на монитор телеметрии	{"method": "TelemetrySubscription", "request_id": "someid"}	
2 Подписка на базовые параметры управления	{"method": "ParametersSubscription", "request_id": "someid"}	
3 Подписка на координаты РЛС	{"method": "LocationSubscription", "request_id": "someid"}	Поддерживается, но не рекомендуется к использованию. В новых разработках вместо него рекомендуется использовать соответствующие поля из метода <i>ParametersSubscription</i> (подписка на базовые параметры управления)
4 Подписка на поток траекторий	{"method": "TrajectorySubscription", "request_id": "someid"}	
5 Команда установки значения параметра	{"method": "SetParameter", "request_id": "someid", "params": {"control_operating": 1, "rotator azimuth speed": 3000}}	
6 Команда отказа от подписки	{"method": "RemoveSubscription", "request_id": "someid", "subscription_req": {"subscription_id": 1}}	Клиентское приложение передаёт subscription_id ранее сформированной подписки, которую необходимо удалить
7 Подписка на пресеты	{"method": "PresetsSubscription", "request_id": "someid"}	
8 Команда изменения состояния пресета	{"method": "SetPresetState", "request_id": "someid", "preset": {"preset_id": "whethear.ini", "state_id": "2"}}	

4.4 В ответ на запрос клиентское приложение получит структуру ответа (Reply) с необязательными полями в соответствии с таблицей 2.

Таблица 2 — Поля ответа Reply

Имя поля	Тип данных поля	Примечание
request_answ	RequestAnsw { “request_id”:”clientrequestid”, “code”: http_status_code, “subscription_id”: subid }	В поле <i>request_id</i> возвращается универсальный идентификатор " someid " из клиентского запроса. В поле <i>code</i> клиенту возвращается статус выполнения команды. В поле <i>subscription_id</i> возвращается номер созданной подписки
subscription_id	integer	Передаётся при отправке сообщений подписки (отказа от подписки) в сторону клиента
telemetry	TelemetryData	Отправка клиенту сообщений с телеметрией
params	ParametersData	Отправка клиенту сообщений с основными параметрами
location	LocationData	Отправка сообщений с координатами РЛС
trajectory	TrajectoryData	Отправка сообщений с траекториями
presets	[PresetDescription]	Массив с описанием пресетов

4.5 Порядок взаимодействия:

1. Клиентское приложения после установления соединения отправляет на сервер команду (запрос) Request, в которой запрашивает нужный тип данных (установка параметров либо подписка на нужные), см. таблицу 1.

2. В ответ клиентское приложение получает ответ (Reply), в котором будет только поле request_answ (формат структуры данных см. в таблице 2).

3. Если клиентское приложение осуществило подписку на какие-то данные, то сервер начнет отправлять в сокет сообщения типа Reply, у которых есть поле subscription_id и запрошенный тип данных. Например, если клиентское приложение отправлял команду {"method": "TelemetrySubscription", "request_id": "38"}, то он будет получать Reply с полем telemetry. Другие поля (request_answ, params и пр.) будут отсутствовать.

4.6 Программа содержит примеры использования API. Для запуска примеров SDK следует:

- для операционной системы Windows установить Python;
- для операционной системы Ubuntu выполнить команды

```
sudo apt update
sudo apt install python3
sudo apt install python3-pip
sudo apt install python3-venv python3-dev
```

Инструкция по запуску примеров приведена в файле *sdk\py\readme.md* из каталога установки Программы.

Приложение А
(обязательное)

Описание команд, ответов и структур данных

А.1 Подписка на монитор телеметрии

Клиентское приложение должно сформировать команду Request, с именем метода *TelemetrySubscription* в соответствии с таблицей А.1.1.

Структура ответа Reply Программы на команду *TelemetrySubscription* — таблица А.1.2.

Структура данных Reply, отправляемая в сокет в случае успешно созданной подписки, описана в таблице А.1.3. Эти данные передаются сразу после создания подписки и содержат полный перечень данных телеметрии. Далее эта структура данных присылается клиенту только при изменении одного из параметров. В таком случае в структуре передаются только измененные данные. Таким образом, все поля структуры являются необязательными. Сама структура данных телеметрии *TelemetryData* описана в таблице А.1.4.



Выделены ключевые параметры телеметрии (см. рисунок А.1).

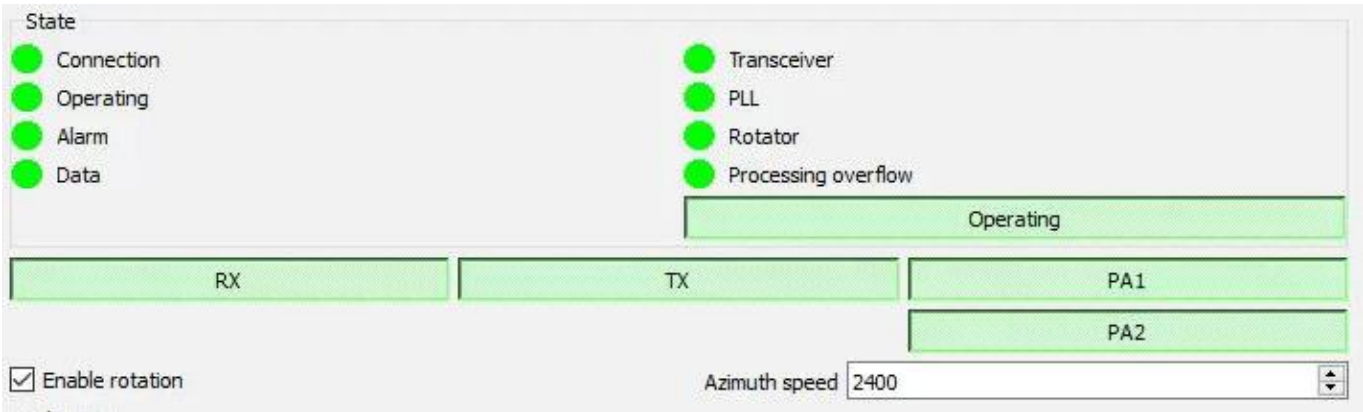


Рисунок А.1 — Восемь индикаторов статуса и ключевые элементы управления

Таблица А.1.1 — Структура данных *TelemetrySubscription*

Имя параметра	Тип параметра	Описание
method	text	TelemetrySubscription
request_id	text	Клиент передаёт уникальный идентификатор запроса, чтобы найти ответ на этот запрос

Таблица А.1.2 — Структура ответа Reply на команду *TelemetrySubscription*

Имя параметра	Тип параметра	Описание
request_answ	RequestAnsw { “request_id”: “clientrequestid”, “code”: http_status_code, “subscription_id”: subid }	В поле <i>request_id</i> возвращается универсальный идентификатор “someid” из клиентского запроса. В поле <i>code</i> клиенту возвращается статус выполнения команды. В поле <i>subscription_id</i> возвращается номер созданной подписки
subscription_id	Не передаётся	В ответе на команду <i>TelemetrySubscription</i> это поле не передаётся
telemetry	Не передаётся	В ответе на команду <i>TelemetrySubscription</i> это поле не передаётся
params	Не передаётся	В ответе на команду <i>TelemetrySubscription</i> это поле не передаётся
location	Не передаётся	В ответе на команду <i>TelemetrySubscription</i> это поле не передаётся
trajectory	Не передаётся	В ответе на команду <i>TelemetrySubscription</i> это поле не передаётся
presets	Не передаётся	В ответе на команду <i>TelemetrySubscription</i> это поле не передаётся

Таблица А.1.3 — Структура данных Reply, отправляемая в Websocket в случае успешно созданной подписки *TelemetrySubscription*

Имя параметра	Тип параметра	Описание
subscription_id	integer	Номер подписки
request_answ	Не передаётся	При передаче данных это поле не используется
params	Не передаётся	При передаче данных это поле не используется
location	Не передаётся	При передаче данных это поле не используется
telemetry	TelemetryData	Данные телеметрии — таблица А.1.4
trajectory	Не передаётся	При передаче данных это поле не используется
presets	Не передаётся	При передаче данных это поле не используется

Таблица А.1.4 — Структура данных TelemetryData. Все поля необязательные

Имя	Тип	Описание
data_stream_status	string enum: “ok”, “warning”, “error”, “not_connected”	 Наличие потока в интерфейсе передачи данных: “ok” — поток есть; “error” — потока нет
recv_speed	double	Скорость потока данных, Мбит/с
fps	integer	Скорость потока данных, сообщений в секунду

Имя	Тип	Описание
lost_frames_percent	double	Процент потерянных сообщений
azimuth_units	integer	Текущий азимут, ед. (от 0 до 4095)
azimuth_angle	double	Текущий азимут, град.
tilt_angle	double	Мгновенное значение угла наклона антенны, град. Ноль — горизонт. Положительное направление — выше горизонта. То же, что <i>rotator_elevation</i> , но получен из интерфейса передачи данных
rotation_speed_rpm	double	Скорость вращения антенны, об./мин
local_time	string	Локальное время, от момента включения (из интерфейса передачи данных). Пример: “04:30:35.537”
lost_intervals	double	Процент потерянных когерентных интервалов
control_connection_state	string enum: “ok”, “warning”, “error”, “not_connected”	 Состояние соединения по интерфейсу управления: “ok” — соединение есть; “error” — соединения нет
preproc_operating_set	string enum: “ok”, “warning”, “error”, “not_connected”	Установленное состояние РЛС: “ok” — включено; “error” — отключено; “not_connected” — нет соединения
preproc_operating_state	string enum: “ok”, “warning”, “error”, “not_connected”	 Фактическое состояние РЛС: “ok” — включено; “error” — отключено; “not_connected” — нет соединения
preproc_serial_number	string	Серийный номер. Пример: “250703”
preproc_version	string	Версия встроенного программного обеспечения. Пример: “79.00.01.75”
preproc_revision	string	Код исполнения. Пример: “10200VS”
preproc_device_time	string	Локальное время, от момента включения (из управляющего интерфейса). Пример: “04:30:35.537”
preproc_temperature	float	Температура препроцессора, град.
preproc_total_time	string	Общее время наработки, ЧЧ:ММ Пример: “1223:02”

Имя	Тип	Описание
transceiver_interface_status	string enum: “ok”, “warning”, “error”, “not_connected”	 Состояние интерфейса к трансиверу: “ok” — ответы есть; “error” — ответов нет; “not_connected” — соединения нет
transceiver_failure	string enum: “ok”, “warning”, “error”, “not_connected”	 Авария трансивера по скажности или длительности импульсов: “ok” — норма; “error” — авария; “not_connected” — соединения нет
transceiver_pll_status	string enum: “ok”, “warning”, “error”, “not_connected”	 Статус петли ФАПЧ: “ok” — захват есть; “error” — захвата нет; “not_connected” — соединения нет
transceiver_current	double	Ток трансивера, показания датчика в вольтах
transceiver_temperature	double	Температура трансивера, град.
transceiver_power	double	Мощность трансивера, показания датчика в вольтах
pa1_current	double	Ток усилителя мощности 1, показания датчика в вольтах
pa1_temperature	double	Температура усилителя мощности 1, град.
pa1_power	double	Мощность усилителя мощности 1, показания датчика в вольтах
pa2_current	double	Ток усилителя мощности 2, показания датчика в вольтах
pa2_temperature	double	Температура усилителя мощности 2, град.
pa2_power	double	Мощность усилителя мощности 2, показания датчика в вольтах
rotator_status	string enum: “ok”, “warning”, “error”, “not_connected”	 Состояние интерфейса к контроллеру электропривода: “ok” — ответы есть; “error” — ответов нет; “not_connected” — соединения нет
rotator_azimuth_speed	string	Текущая скорость вращения по азимуту, ед.
rotator_elevation	double	Угол наклона антенны, град. Ноль — горизонт. Положительное направление — выше горизонта. То же, что <i>tilt_angle</i> , но получен из интерфейса управления

Имя	Тип	Описание
rotator_ext_temperature_1	double	Температура на внешнем датчике 1 электропривода, град.
rotator_ext_temperature_2	double	Температура на внешнем датчике 2 электропривода, град.
processing_overflow	string enum: “ok”, “warning”, “error”, “not_connected”	 Переполнение очереди сигнальной обработки: “ok” — норма; “warning” — предупреждение; “error” — переполнение; “not_connected” — нет потока данных
detector_stat	struct	Статистика обнаружения
points_after_cfar	integer	Количество точек после детектора
points_after_cband	integer	Количество точек после детектора и фильтра центральной полосы
clusters_stat	struct	Статистика кластеризации
total	integer	Всего
size_filtered	integer	Отфильтровано по размеру
comp_filtered	integer	Отфильтровано по компенсационному каналу
clutter_filtered	integer	Отфильтровано клаттером
sky_filtered	integer	Отфильтровано по высоте
unfiltered	integer	Не отфильтровано
tracks_stat	struct	Статистика автосопровождения
total	integer	Всего
reliable	integer	Надежных

Пример сообщения с подпиской на монитор телеметрии с полным набором данных:

```
{
  "subscription_id": 2,
  "telemetry": {
    'recv_speed': '74.57 mbit/s',
    'fps': 250,
    'lost_frames_percent': '0.00 %',
    'azimuth_units': 2222,
    'azimuth_angle': 195.29296875,
    'tilt_angle': 5.82,
    'rotation_speed_rpm': 25.634766166149,
    'local_time': '72:55:49.966',
    'lost_intervals': '0.00 %',
```

```
'control_connection_state': 'ok',
'preproc_operating_set': 'ok',
'preproc_operating_state': 'ok',
'preproc_serial_number': '250799',
'preproc_version': '79.0.1.77',
'preproc_revision': '10200TS',
'preproc_device_time': '72:55:49.862',
'preproc_temperature': 45.863960266113,
'preproc_total_time': '94:19 minutes',
'transceiver_interface_status': 'ok',
'transceiver_failure': 'ok',
'transceiver_pll_status': 'ok',
'transceiver_current': 0.97904296875,
'transceiver_temperature': 36.024992382103,
'transceiver_power': 0.004028320312,
'pa1_current': 0.1843359375,
'pa1_temperature': 30.337547049474,
'pa1_power': 0.124072265625,
'pa2_current': 0.23009765625,
'pa2_temperature': 33.355236971142,
'pa2_power': 0.120849609375,
'rotator_status': 'ok',
'rotator_azimuth_speed': 3000,
'rotator_elevation': 5.765625,
'processing_overflow': 'ok',
'detector_stat':
{
    'points_after_cfar': 0,
    'points_after_cband': 0
},
'clusters_stat':
{
    'total': 0,
    'size_filtered': 0,
    'comp_filtered': 1,
    'clutter_filtered': 0,
```

```
    'sky_filtered': 0,  
    'unfiltered': 0  
  },  
  'tracks_stat':  
  {  
    'total': 8,  
    'reliable': 0  
  }  
}  
}
```

Пример сообщения с несколькими измененными параметрами:

```
{  
  "subscription_id": 2,  
  "telemetry": {  
    "fps": null,  
    "azimuth_units": 3666,  
    "azimuth_angle": null,  
    "transceiver_current": null,  
    "transceiver_temperature": null,  
    "transceiver_power": null,  
    "pa1_current": null,  
    "pa1_temperature": null,  
    "pa1_power": null,  
    "pa2_current": null,  
    "pa2_temperature": null,  
    "pa2_power": null,  
    "rotator_azimuth_speed": null,  
    "rotator_elevation": null  
  }  
}
```


А.2 Подписка на базовые параметры управления

Клиентское приложение должно сформировать команду Request с именем метода *ParametersSubscription* в соответствии с таблицей А.2.1.

Структура ответа Reply Программы на команду *ParametersSubscription* — таблица А.2.2.

Структура данных Reply, отправляемая в Websocket в случае успешно созданной подписки — таблица А.2.3. Эти данные передаются сразу после создания подписки и содержат полный перечень базовых параметров управления. Далее эта структура данных присылается клиенту только при изменении одного из параметров. В таком случае в структуре передаются только измененные данные. Таким образом, все поля структуры являются необязательными. Сама структура данных телеметрии *ParametersData* описана в таблице А.2.4.

Таблица А.2.1 — Структура данных Request с методом *ParametersSubscription*

Имя параметра	Тип параметра	Описание
method	text	ParametersSubscription
request_id	text	Клиент передаёт уникальный идентификатор запроса, чтобы найти ответ на этот запрос

Таблица А.2.2 — Структура ответа Reply на команду *ParametersSubscription*

Имя параметра	Тип параметра	Описание
request_answ	RequestAnsw <pre>{ "request_id": "clientrequestid", "code": http_status_code, "subscription_id": subid }</pre>	В поле <i>request_id</i> возвращается универсальный идентификатор " <i>someid</i> " из клиентского запроса. В поле <i>code</i> клиенту возвращается статус выполнения команды. В поле <i>subscription_id</i> возвращается номер созданной подписки
subscription_id	Не передаётся	В ответе на команду <i>ParametersSubscription</i> это поле не передаётся
telemetry	Не передаётся	В ответе на команду <i>ParametersSubscription</i> это поле не передаётся
params	Не передаётся	В ответе на команду <i>ParametersSubscription</i> это поле не передаётся
location	Не передаётся	В ответе на команду <i>ParametersSubscription</i> это поле не передаётся
trajectory	Не передаётся	В ответе на команду <i>ParametersSubscription</i> это поле не передаётся
presets	Не передаётся	В ответе на команду <i>ParametersSubscription</i> это поле не передаётся

Таблица А.2.3 — Структура данных Reply, отправляемая в Websocket в случае успешно созданной подписки *ParametersSubscription*

Имя параметра	Тип параметра	Описание
subscription_id	integer	Номер подписки
request_answ	Не передаётся	При передаче данных это поле не используется.
params	ParametersData	Базовые параметры управления — таблица А.2.4
location	Не передаётся	При передаче данных это поле не используется
telemetry	Не передаётся	При передаче данных это поле не используется
trajectory	Не передаётся	При передаче данных это поле не используется
presets	Не передаётся	При передаче данных это поле не используется

Таблица А.2.4 — Структура данных ParametersData, все поля необязательные

Имя	Тип	Описание
radar_location_latitude	double	Координаты точки стояния РЛС, широта, град.
radar_location_longitude	double	Координаты точки стояния РЛС, долгота, град.
radar_location_azimuth	double	Координаты точки стояния РЛС, азимут (поправка условного нуля относительно севера), град.
radar_location_height	double	Координаты точки стояния РЛС, высота относительно условной плоскости, м
control_operating	bool	Состояние РЛС (true — включено, false — отключено)
recording_dump_writer	bool	Запись сырых данных (true — включена, false — отключена)
recording_track_writer	bool	Запись траекторий (true — включена, false — отключена)
transceiver_strobe_tx	bool	Состояние питания передатчика (true — включено, false — отключено)
transceiver_strobe_pa1	bool	Состояние питания усилителя мощности 1 (true — включено, false — отключено)
transceiver_strobe_pa2	bool	Состояние питания усилителя мощности 2 (true — включено, false — отключено)
transceiver_strobe_rx	bool	Состояние питания малошумящих усилителей (true — включено, false — отключено)
rotator_azimuth_speed	integer	Текущая скорость вращения по азимуту, ед.
rotator_azimuth_control	bool	Управление электроприводом (true — включено, false — отключено)
rotator_elevation_control	string enum: “stop”, “move_up”, “move_down”	Состояние угломестного привода (stop — останов, move_up — движение вверх, move_down — движение вниз)
debug_output_all	bool	Выводить в консоль все отладочные сообщения

Имя	Тип	Описание
debug_output_ws	bool	Выводить в консоль отладочные сообщения модуля ws (websocket, программный интерфейс приложения)
debug_output_control_connection	bool	Выводить в консоль отладочные сообщения модуля control_connection (интерфейс управления)
debug_output_external_tracks	bool	Выводить в консоль отладочные сообщения модуля external_tracks (проигрыватель траекторий)
debug_output_recording	bool	Выводить в консоль отладочные сообщения модуля recording (запись исходных данных и траекторий)
debug_output_parser	bool	Выводить в консоль отладочные сообщения модуля parser (декодер интерфейса передачи данных)
debug_output_detector	bool	Выводить в консоль отладочные сообщения модуля detector (алгоритм обнаружения)
debug_output_clutter	bool	Выводить в консоль отладочные сообщения модуля clutter (алгоритм фильтра пассивных помех)
debug_output_tracking	bool	Выводить в консоль отладочные сообщения модуля tracking (алгоритм автосопровождения)
preprocessor_disabled_sector1	bool	Отключение излучение в секторе 1 (true — излучение отключено, false — излучение включено)
preprocessor_start_azimuth1	double	Начальный азимут сектора 1, град.
preprocessor_end_azimuth1	double	Конечный азимут сектора 1, град.
preprocessor_disabled_sector2	bool	Отключение излучение в секторе 2 (true — излучение отключено, false — излучение включено)
preprocessor_start_azimuth2	double	Начальный азимут сектора 2, град.
preprocessor_end_azimuth2	double	Конечный азимут сектора 2, град.
preprocessor_disabled_sector3	bool	Отключение излучение в секторе 3 (true — излучение отключено, false — излучение включено)
preprocessor_start_azimuth3	double	Начальный азимут сектора 3, град.
preprocessor_end_azimuth3	double	Конечный азимут сектора 3, град.
parser_phase0	double	Фаза приемного канала 1
parser_phase1	double	Фаза приемного канала 2
parser_phase2	double	Фаза приемного канала 3
detector_azimuth_masking_begin_deg	double	Маскирование алгоритма обнаружения в азимутальном секторе (начальный азимут), град.
detector_azimuth_masking_end_deg	double	Маскирование алгоритма обнаружения в азимутальном секторе (конечный азимут), град.
detector_coherent_size	integer	Размер когерентного интервала
detector_coherent_overlap	integer	Перекрытие когерентных интервалов
detector_lost_intervals_filter	integer	Допустимые потери внутри когерентного интервала, %
detector_cfar_threshold	double	Порог детектора, дБ
detector_cband_threshold	double	Порог фильтра центральной полосы, м/с

Имя	Тип	Описание
detector_min_range	double	Порог минимальной дальности, м
detector_cluster_range_gap	integer	Кластеризация: максимальный зазор внутри кластера по дальности, ед.
detector_cluster_doppler_gap	integer	Кластеризация: максимальный зазор внутри кластера по скорости, ед.
detector_cluster_azimuth_gap	double	Кластеризация: максимальный зазор внутри кластера по азимуту, град.
detector_cluster_min_range_width	double	Кластеризация: минимальный размер кластера по дальности, м
detector_cluster_max_range_width	double	Кластеризация: максимальный размер кластера по дальности, м
detector_cluster_min_doppler_width	integer	Кластеризация: минимальный размер кластера по скорости, ед.
detector_cluster_max_doppler_width	integer	Кластеризация: максимальный размер кластера по скорости, ед.
detector_sky_filter	double	Порог максимальной высоты, м
detector_comp_threshold	double	Порог компенсационного канала, дБ
classification_reliable_points_count	integer	Классификация: число точек в надежном треке
classification_max_ground_altitude	double	Классификация: максимальная высота наземной цели, м
classification_max_human_speed	double	Классификация: максимальная скорость пешехода, м/с
classification_max_noise_rcs	double	Классификация: максимальная ЭПР шума, дБ (кв. м)
classification_max_birds_rcs	double	Классификация: максимальная ЭПР птицы, дБ (кв. м)
classification_max_pedestr_rcs	double	Классификация: максимальная ЭПР пешехода, дБ (кв. м)
clutter_range_cell_size	double	Фильтр пассивных помех: размер ячейки по дальности, м
clutter_doppler_cell_size	integer	Фильтр пассивных помех: размер ячейки по радиальной скорости, ед.
clutter_azimuth_cell_size	double	Фильтр пассивных помех: размер ячейки по азимуту, град.
clutter_history_size	integer	Фильтр пассивных помех: длина истории (скользящего окна)
clutter_threshold	integer	Фильтр пассивных помех: порог от длины истории
tracking_track_lifetime	double	Автосопровождение: время жизни траектории, с
tracking_max_range_deviation	double	Автосопровождение: максимальное отклонение по дальности, м
tracking_max_speed_deviation	double	Автосопровождение: максимальное отклонение по скорости, м
tracking_max_azimuth_deviation	double	Автосопровождение: максимальное отклонение по азимуту, м

Имя	Тип	Описание
tracking_max_penalty	double	Автосопровождение: максимальный штраф, м
tracking_clear_timeout	integer	Автосопровождение: время удаления «зависших» треков, с
tracking_extrapolation_timeout	integer	Автосопровождение: шаг экстраполяции, мс (0 — экстраполяции траекторий нет)
att_tx_if	double	Трансивер: ослабление аттенюатора промежуточной частоты передатчика, дБ
att_tx_rf	integer	Трансивер: ослабление аттенюатора радиочастоты передатчика, дБ
att_rx1_if	double	Трансивер: ослабление аттенюатора промежуточной частоты приемника (канал 1), дБ
att_rx1_rf	integer	Трансивер: ослабление аттенюатора радиочастоты приемника (канал 1), дБ
att_rx2_if	double	Трансивер: ослабление аттенюатора промежуточной частоты приемника (канал 2), дБ
att_rx2_rf	integer	Трансивер: ослабление аттенюатора радиочастоты приемника (канал 2), дБ
att_rx3_if	double	Трансивер: ослабление аттенюатора промежуточной частоты приемника (канал 3), дБ
att_rx3_rf	integer	Трансивер: ослабление аттенюатора радиочастоты приемника (канал 3), дБ
att_rx4_if	double	Трансивер: ослабление аттенюатора промежуточной частоты приемника (канал 4), дБ
att_rx4_rf	integer	Трансивер: ослабление аттенюатора радиочастоты приемника (канал 4), дБ
api_trj_imitated	bool	Отправляемые в API классы: Имитированная цель
api_trj_person	bool	Отправляемые в API классы: Человек
api_trj_car	bool	Отправляемые в API классы: Автомобиль
api_trj_drone	bool	Отправляемые в API классы: Дрон
api_trj_plane	bool	Отправляемые в API классы: Самолет
api_trj_birds	bool	Отправляемые в API классы: Птицы
api_trj_boat	bool	Отправляемые в API классы: Судно
api_trj_noise	bool	Отправляемые в API классы: Шум
api_trj_target	bool	Отправляемые в API классы: Неопознанная цель

Пример сообщения с подпиской на базовые параметры управления:

```
{
  "subscription_id": 1,
  "params":
  {
    "control_operating": false,
    "transceiver_strobe_tx": false,
    "transceiver_strobe_pa1": false,
    "transceiver_strobe_pa2": false,
    "transceiver_strobe_rx": false,
    "rotator_azimuth_speed": 0,
    "rotator_elevation_control": 'stop',
    "rotator_azimuth_control": false
    "debug_output_all": false,
    "debug_output_ws": false,
    "debug_output_control_connection": false,
    "debug_output_external_tracks": false,
    "debug_output_recording": false,
    "debug_output_parser": false,
    "debug_output_detector": false,
    "debug_output_clutter": false,
    "debug_output_tracking": false
  }
}
```

А.3 Подписка на координаты РЛС

Поддерживается, но не рекомендуется к использованию. В новых разработках вместо него рекомендуется использовать соответствующие поля из метода `ParametersSubscription` (подписка на базовые параметры управления), см. раздел А.2.

Клиентское приложение должно сформировать команду `Request` с именем метода `LocationSubscription` в соответствии с таблицей А.3.1.

Структура ответа `Reply` Программы на команду `LocationSubscription` — таблица А.3.2.

Структура данных `Reply`, отправляемая в `Websocket` в случае успешно созданной подписки — таблица А.3.3. Эти данные передаются сразу после создания подписки и содержат полный перечень координат РЛС. Далее эта структура данных присылается клиенту только при изменении одного из параметров. В таком случае в структуре передаются только измененные данные. Таким образом, все поля структуры являются необязательными. Сама структура данных с координатами РЛС `LocationData` описана в таблице А.3.4.

Таблица А.3.1 — Структура данных `Request` с методом `LocationSubscription`

Имя параметра	Тип параметра	Описание
method	text	LocationSubscription
request_id	text	Клиент передаёт уникальный идентификатор запроса, чтобы найти ответ на этот запрос

Таблица А.3.2 — Структура ответа `Reply` на команду `LocationSubscription`

Имя параметра	Тип параметра	Описание
request_answ	RequestAnsw { "request_id": "clientrequestid", "code": http_status_code, "subscription_id": subid }	В поле <code>request_id</code> возвращается универсальный идентификатор " someid " из клиентского запроса. В поле <code>code</code> клиенту возвращается статус выполнения команды. В поле <code>subscription_id</code> возвращается номер созданной подписки
subscription_id	Не передаётся	В ответе на команду <code>LocationSubscription</code> это поле не передаётся
telemetry	Не передаётся	В ответе на команду <code>LocationSubscription</code> это поле не передаётся
params	Не передаётся	В ответе на команду <code>LocationSubscription</code> это поле не передаётся
location	Не передаётся	В ответе на команду <code>LocationSubscription</code> это поле не передаётся
trajectory	Не передаётся	В ответе на команду <code>LocationSubscription</code> это поле не передаётся

Имя параметра	Тип параметра	Описание
presets	Не передаётся	В ответе на команду <i>LocationSubscription</i> это поле не передаётся

Таблица А.3.3 — Структура данных Reply, отправляемая в Websocket в случае успешно созданной подписки *LocationSubscription*

Имя параметра	Тип параметра	Описание
subscription_id	integer	Номер подписки
request_answ	Не передаётся	При передаче данных это поле не используется
params	Не передаётся	При передаче данных это поле не используется
location	LocationData	Координаты РЛС — таблица А.3.4
telemetry	Не передаётся	При передаче данных это поле не используется
trajectory	Не передаётся	При передаче данных это поле не используется
presets	Не передаётся	При передаче данных это поле не используется

Таблица А.3.4 — Структура данных LocationData, все поля необязательные

Имя	Тип	Описание
latitude	double	Широта, град.
longitude	double	Долгота, град.
azimuth	double	Азимут (поправка условного нуля относительно севера), град.
height	double	Высота относительно условной плоскости, м

Пример сообщения на подписку координат РЛС:

```
{
  "subscription_id": 1,
  "location":
  {
    "latitude": 56.513771057129,
    "longitude": 34.970371246338,
    "azimuth": 0.0,
    "height": 10.0
  }
}
```


А.4 Подписка на поток траекторий

Клиентское приложение должно сформировать команду Request с именем метода *TrajectorySubscription* в соответствии с таблицей А.4.1.

Структура ответа Reply Программы на команду *TrajectorySubscription* — таблица А.4.2.

Структура данных Reply, отправляемая в Websocket в случае успешно созданной подписки — таблица А.4.3. Эти данные передаются сразу после создания подписки и содержат новые точки траекторий. В первом сообщении клиент получает все точки всех треков, которые существуют на момент подписки. Затем клиент получает информацию о новых точках траекторий или об удаленных траекториях. Структура данных с точками траекторий *TrajectoryData* описана в таблице А.4.4.

Порядок использования Программы для просмотра тестовых траекторий изложен в Приложении Б.

Таблица А.4.1 — Структура данных Request с методом *TrajectorySubscription*

Имя параметра	Тип параметра	Описание
method	text	<i>TrajectorySubscription</i>
request_id	text	Клиент передаёт уникальный идентификатор запроса, чтобы найти ответ на этот запрос

Таблица А.4.2 — Структура ответа Reply на команду *TrajectorySubscription*

Имя параметра	Тип параметра	Описание
request_answ	RequestAnsw <pre>{ "request_id": "clientrequestid", "code": http_status_code, "subscription_id": subid }</pre>	В поле <i>request_id</i> возвращается универсальный идентификатор " <i>someid</i> " из клиентского запроса. В поле <i>code</i> клиенту возвращается статус выполнения команды. В поле <i>subscription_id</i> возвращается номер созданной подписки
subscription_id	Не передаётся	В ответе на команду <i>TrajectorySubscription</i> это поле не передаётся
telemetry	Не передаётся	В ответе на команду <i>TrajectorySubscription</i> это поле не передаётся
params	Не передаётся	В ответе на команду <i>TrajectorySubscription</i> это поле не передаётся
location	Не передаётся	В ответе на команду <i>TrajectorySubscription</i> это поле не передаётся
trajectory	Не передаётся	В ответе на команду <i>TrajectorySubscription</i> это поле не передаётся
presets	Не передаётся	В ответе на команду <i>TrajectorySubscription</i> это поле не передаётся

Таблица А.4.3 — Структура данных Reply, отправляемая в Websocket в случае успешно созданной подписки *TrajectorySubscription*

Имя параметра	Тип параметра	Описание
subscription_id	integer	Номер подписки
request_answ	Не передаётся	При передаче данных это поле не используется
params	Не передаётся	При передаче данных это поле не используется
location	Не передаётся	При передаче данных это поле не используется
telemetry	Не передаётся	При передаче данных это поле не используется
trajectory	TrajectoryData	Информация о треках — таблица А.4.4.
presets	Не передаётся	При передаче данных это поле не используется

Таблица А.4.4 — Структура данных *TrajectoryData*

Имя	Тип	Описание
time	string	Время кадра
removed_tracks	array [int]	Массив идентификаторов треков, которые больше не повторяются
remove_all	bool	Признак удаления всех живых треков
new_points	Array PointInfo	Новые точки траектории, структура <i>PointInfo</i> в таблице А.4.5

Таблица А.4.5 — Структура данных *PointInfo*

Имя	Тип	Описание
track_id	int	ID трека
points_count	int	Физический смысл — число точек трека (количество обнаружений), параметр определяет «надежность» трека (<i>is_reliable</i>)
time	string	Время обнаружения в формате «уууу-ММ-dd hh:mm:ss»
geo_position	struct	Геокоординаты
latitude *	double	Широта, град.
longitude *	double	Долгота, град
altitude	double	Высота относительно условной плоскости, м. К использованию рекомендуется значение <i>tracker_info.altitude</i>
local_position	struct	Координаты в местной системе
range *	float	Наклонная дальность, м
azimuth *	float	Азимут относительно условного «нуля» (увеличение по часовой стрелке), град.
gps_azimuth *	float	Азимут относительно севера (увеличение по часовой стрелке), град.

Имя	Тип	Описание
elevation	float	Угол места, град., от нормали к полотну антенны. Диапазон от минус 90 до плюс 90. Положительное направление — вверх
altitude	float	То же, что geo_position.altitude. Вычисляется с использованием значений elevation и tilt_angle, см. таблицу А.1.4. К использованию рекомендуется значение <i>tracker_info.altitude</i>
radial_speed	float	Радиальная скорость, м/с
class_name	enum	“imitated”  “target”  “person”  “car”  “boat”  “birds”  “drone”  “plane”  “noise” 
reliability	double	Вероятность надежности траектории, от 0 до 1
is_manual	bool	Цель взята на ручное сопровождение. Рекомендуется к использованию для выбора приоритета
is_reliable	bool	Обобщенный показатель надежности траектории
user_info	string	Служебный текст

Имя	Тип	Описание
is_extrapolated *	bool	Точка траектории экстраполирована
power_data	struct	Энергетические характеристики
points_count	int	Число точек в кластере
range_width_m	int	Размер кластера по дальности, м
doppler_width	int	Размер кластера по скорости (доплеровскому спектру), гармоник
azimuth_width	int	Размер кластера по азимуту, когерентных интервалов
target_power_db	float	Мощность центра тяжести кластера в суммарном канале, дБ
noise_power_db	float	Средняя мощность шума, дБ
compensation_power_db	float	Мощность центра тяжести кластера в компенсационном канале, дБ
energy_db	float	Энергия кластера, сумма мощностей всех точек кластера, дБ
snr	float	Отношение сигнала к шуму для центра тяжести кластера, дБ
snr_cluster	float	Отношение сигнала к шуму для всего кластера, дБ
rcs	float	Оценка эффективной площади рассеяния для центра тяжести кластера, дБ (м ²) К использованию рекомендуется значение <i>tracker_info.rcs</i>
rcs_cluster	float	Оценка эффективной площади рассеяния для всего кластера, дБ (м ²) К использованию рекомендуется значение <i>tracker_info.rcs</i>
tracker_info	struct	Траекторные данные
azimuthal_speed	float	Угловая скорость, град/с
tangential_speed	float	Тангенциальная скорость, м/с
radial_speed	float	Радиальная скорость (по треку), м/с
full_speed	float	Полная скорость, м/с
altitude	float	Высота относительно условной плоскости, сглаженная по треку, м. Рекомендуется к использованию вместо <i>geo_position.altitude</i> и <i>local_position.altitude</i>
rcs	float	Оценка эффективной площади рассеяния для центра тяжести кластера, сглаженная по треку, дБ (м ²) Рекомендуется к использованию вместо <i>power_data.rcs</i>
rcs_cluster	float	Оценка эффективной площади рассеяния для всего кластера, сглаженная по треку, дБ (м ²) К использованию рекомендуется значение <i>tracker_info.rcs</i>

Имя	Тип	Описание
penalty_track_id	int	Идентификатор ближайшей связываемой траектории
penalty_rotation_counts	int	Число оборотов от последнего подтверждения ближайшей связываемой траектории
penalty_speed_deviation	float	Отклонение по полной скорости от прогнозируемого положения ближайшей связываемой траектории, в метрах
penalty_range_deviation	float	Отклонение по дальности от прогнозируемого положения ближайшей связываемой траектории, в метрах
penalty_azimuth_deviation	float	Отклонение по азимуту от прогнозируемого положения ближайшей связываемой траектории, в метрах
penalty	float	Полный штраф от ближайшей связываемой траектории, в метрах
* При (is_extrapolated = true) экстраполируются следующие данные: <i>geo_position.latitude</i> <i>geo_position.longitude</i> <i>local_position.range</i> <i>local_position.azimuth</i> <i>local_position.gps_azimuth</i> Остальная информация повторяется		

Пример сообщения:

```
{
  "subscription_id": 1,
  "trajectory":
  {
    "time": "2025-09-19 00:17:12",
    "remove_all": false,
    "new_points": [
      {
        "track_id": 1,
        "points_count": 9,
        "geo_position":
        {
          "latitude": 56.523563786289,
          "longitude": 34.953156125523,
          "altitude": 0.0
        },
        "time": "2025-09-19 00:17:12",
```

```
"class_name": "imitated",
"reliability": 1.0
"is_manual": false,
"is_reliable": true,
"user_info": "586",
"is_extrapolated": false,
"local_position":
{
  "range": 200.595043166186,
  "azimuth": 216.995175052256,
  "gps_azimuth": 139.995175052256,
  "elevation": 15.066794914619,
  "altitude": 56.0,
  "radial_speed": 0.0
}
"tracker_info":
{
  "azimuthal_speed": 0.0,
  "tangential_speed": 0.0,
  "radial_speed": 0.0,
  "full_speed": 0.0,
  "altitude": 0.0,
  "rcs": 0.0,
  "rcs_cluster": 0.0,
  'penalty_track_id': 15,
  'penalty_rotation_counts': 2,
  'penalty_speed_deviation': 9.606916144645,
  'penalty_range_deviation': 142.315868430689,
  'penalty_azimuth_deviation': 195.797975615247,
  'penalty': 347.720760190581
}
"power_data":
{
  "points_count": 0,
  "range_width_m": 1,
  "doppler_width": 1,
```

```
    "azimuth_width": 1,  
    "target_power_db": 0.0,  
    "noise_power_db": 0.0,  
    "compensation_power_db": 0.0,  
    "energy_db": 0.0,  
    "snr": 0.0,  
    "snr_cluster": 0.0,  
    "rcs": 0.0,  
    "rcs_cluster": 0.0  
  }  
}  
]  
}  
}
```

A.5 Команда установки значения параметра

Клиентское приложение должно сформировать команду Request с именем метода *SetParameter* в соответствии с таблицей A.5.1.

Структура ответа Reply Программы на команду *SetParameter* — таблица A.5.2. Необходимо проверить код выполнения команды `reply:request_answ::code`. Численные значения кодов возврата соответствуют HTTP-кодам (в частности, код 200 соответствует успешной установке параметров).

Таблица A.5.1 — Структура данных Request с методом *SetParameter*

Имя параметра	Тип параметра	Описание
method	text	<i>SetParameter</i>
request_id	text	Клиент передаёт уникальный идентификатор запроса, чтобы найти ответ на этот запрос
params	ParametersData { "control_operating":bool, "transceiver_strobe_tx": bool, "transceiver_strobe_pa1":bool, "transceiver_strobe_pa2":bool, "transceiver_strobe_rx": bool, "rotator_azimuth_speed":int, "rotator_azimuth_control":bool, "rotator_elevation_control":string, ... }	Клиент передаёт один или несколько параметров, которые необходимо изменить. Параметры, которые не нужно менять, допускается не указывать в структуре params, либо указать их значения как null. Значения параметра "rotator_elevation_control" могут принимать только значения: "stop", "move_up", "move_down". Полный перечень параметров, их наименование, типы и описание см. в таблице A.2.4

Таблица А.5.2 — Структура ответа Reply на команду *SetParameter*

Имя параметра	Тип параметра	Описание
request_answ	RequestAnsw { "request_id": "clientrequestid", "code": http_status_code, "subscription_id": subid }	В поле <i>request_id</i> возвращается универсальный идентификатор " <i>someid</i> " из клиентского запроса. В поле <i>code</i> клиенту возвращается статус установки параметра. В поле <i>subscription_id</i> возвращается номер, который передавался в Request::request_id
subscription_id	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
telemetry	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
params	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
location	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
trajectory	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
presets	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся

Пример запроса установки параметра *rotator_elevation_control*:

```
{
  'method': 'SetParameter',
  'request_id': '56',
  'params': {'rotator_elevation_control': 'MoveDown'}
}
```

Пример запроса установки параметра *transceiver_strobe_rx*:

```
{
  'method': 'SetParameter',
  'request_id': '57',
  'params': {'transceiver_strobe_rx': 1}
}
```

А.6 Команда отказа от подписки

Клиентское приложение должно сформировать команду Request с именем метода *RemoveSubscription* в соответствии с таблицей А.6.1.

Структура ответа Reply Программы на команду *RemoveSubscription* — таблица А.6.2. Необходимо проверить код выполнения команды `reply:request_answ::code`. Численные значения кодов возврата соответствуют HTTP-кодам (в частности, код 200 соответствует успешной установке параметров).

Таблица А.6.1 — Структура данных Request с методом *RemoveSubscription*

Имя параметра	Тип параметра	Описание
method	text	<i>RemoveSubscription</i>
request_id	text	Клиент передаёт уникальный идентификатор запроса, чтобы найти ответ на этот запрос
subscription_req	<i>SubscriptionRequest</i> { “subscription_id”:int }	Клиент передаёт номер подписки, от которой он отказывается.

Таблица А.6.2 — Структура ответа Reply на команду *RemoveSubscription*

Имя параметра	Тип параметра	Описание
request_answ	RequestAnsw { “request_id”:“clientrequestid”, “code”: http_status_code, “subscription_id”: subid }	В поле <i>request_id</i> возвращается универсальный идентификатор “ someid ” из клиентского запроса. В поле <i>code</i> клиенту возвращается статус удаления подписки. В поле <i>subscription_id</i> возвращается номер удаленной подписки
subscription_id	Не передаётся	В ответе на команду <i>RemoveSubscription</i> это поле не передаётся
telemetry	Не передаётся	В ответе на команду <i>RemoveSubscription</i> это поле не передаётся
params	Не передаётся	В ответе на команду <i>RemoveSubscription</i> это поле не передаётся
location	Не передаётся	В ответе на команду <i>RemoveSubscription</i> это поле не передаётся
trajectory	Не передаётся	В ответе на команду <i>RemoveSubscription</i> это поле не передаётся
presets	Не передаётся	В ответе на команду <i>RemoveSubscription</i> это поле не передаётся

Пример запроса на отказ от подписки:

```
{  
  'method': 'RemoveSubscription',  
  'request_id': '566',  
  'subscription_req': {'subscription_id': 2}  
}
```

А.7 Подписка на пресеты

Клиентское приложение должно сформировать команду Request с именем метода *PresetsSubscription* в соответствии с таблицей А.7.1.

Структура ответа Reply Программы на команду *PresetsSubscription* — таблица А.7.2.

Структура данных Reply, отправляемая в Websocket в случае успешно созданной подписки — таблица А.7.3. Эти данные передаются сразу после создания подписки и содержат полный перечень пресетов. Далее эта структура данных присылается клиенту только при любых изменениях пресетов (в том числе активация или деактивация какого-либо состояния). На изменения передаётся полный набор пресетов. Структура данных является массивом из элементов *PresetDescription*. Структура данных *PresetDescription* описана в таблице А.7.4.

Таблица А.7.1 — Структура данных Request с методом *PresetsSubscription*

Имя параметра	Тип параметра	Описание
method	text	PresetsSubscription
request_id	text	Клиент передаёт уникальный идентификатор запроса, чтобы найти ответ на этот запрос

Таблица А.7.2 — Структура ответа Reply на команду *PresetsSubscription*

Имя параметра	Тип параметра	Описание
request_answ	RequestAnsw { “request_id”: “clientrequestid”, “code”: http_status_code, “subscription_id”: subid }	В поле <i>request_id</i> возвращается универсальный идентификатор “ <i>someid</i> ” из клиентского запроса. В поле <i>code</i> клиенту возвращается статус выполнения команды. В поле <i>subscription_id</i> возвращается номер созданной подписки
subscription_id	Не передаётся	В ответе на команду <i>PresetsSubscription</i> это поле не передаётся
telemetry	Не передаётся	В ответе на команду <i>PresetsSubscription</i> это поле не передаётся
params	Не передаётся	В ответе на команду <i>PresetsSubscription</i> это поле не передаётся
location	Не передаётся	В ответе на команду <i>PresetsSubscription</i> это поле не передаётся
trajectory	Не передаётся	В ответе на команду <i>PresetsSubscription</i> это поле не передаётся
presets	Не передаётся	В ответе на команду <i>PresetsSubscription</i> это поле не передаётся

Таблица А.7.3 — Структура данных Reply, отправляемая в Websocket в случае успешно созданной подписки *PresetsSubscription*

Имя параметра	Тип параметра	Описание
subscription_id	integer	Номер подписки
request_answ	Не передаётся	При передаче данных это поле не используется
params	Не передаётся	При передаче данных это поле не используется
location	Не передаётся	При передаче данных это поле не используется
telemetry	Не передаётся	При передаче данных это поле не используется
trajectory	Не передаётся	При передаче данных это поле не используется
presets	[PresetDescription]	Массив с элементами PresetDescription – таблица А.7.4

Таблица А.7.4 — Структура данных *PresetDescription*, все поля необязательные

Имя	Тип	Описание
name	text	Имя пресета
description	text	Описание пресета
id	text	Уникальный идентификатор данного пресета
states	[PresetStateDescription]	Массив с элементами PresetStateDescription – таблица А.7.5

Таблица А.7.5 — Структура данных *PresetStateDescription*, все поля необязательные

Имя	Тип	Описание
name	text	Имя состояния пресета
description	text	Описание состояния пресета
id	text	Уникальный идентификатор состояния пресета
activated	bool	Флаг, показывающий активировано ли данное состояние

Пример сообщения на подписку пресетов:

```
{
  "subscription_id": 4,
  "presets": [
    {
      "name": "Weather",
      "description": "Central band and clutter cell preset",
      "id": "weather.ini",
      "states": [
        {
          "name": "No wind",
          "description": "1 m/s",
          "id": "0",
          "activated": false
        },
        {
          "name": "Light wind",
          "description": "5 m/s",
          "id": "1",
          "activated": false
        },
        {
          "name": "Strong wind",
          "description": "10 m/s",
          "id": "2",
          "activated": false
        }
      ]
    },
    {
      "name": "Weather2",
      "description": "Central band and clutter cell preset",
      "id": "weather2.ini",
      "states": [
        {
          "name": "No wind2",
```

```
    "description": "1 m/s",
    "id": "0",
    "activated": false
  },
  {
    "name": "Light wind2",
    "description": "5 m/s",
    "id": "1",
    "activated": true
  },
  {
    "name": "Strong wind2",
    "description": "10 m/s",
    "id": "2",
    "activated": false
  }
]
}
]
```

А.8 Команда изменения состояния пресета

Клиентское приложение должно сформировать команду Request с именем метода *SetPresetState* в соответствии с таблицей А.8.1.

Структура ответа Reply Программы на команду *SetPresetState* — таблица А.8.2. Необходимо проверить код выполнения команды *reply:request_answ::code*. Численные значения кодов возврата соответствуют HTTP-кодам (в частности, код 200 соответствует успешной активации состояния пресета).

Таблица А.8.1 — Структура данных Request с методом *SetPresetState*

Имя параметра	Тип параметра	Описание
method	text	<i>SetPresetState</i>
request_id	text	Клиент передаёт уникальный идентификатор запроса, чтобы найти ответ на этот запрос
preset	<i>PresetState</i> { “preset_id”:text, “state_id”: text }	Клиент передаёт в атрибуте preset структуру с двумя атрибутами: preset_id — уникальный идентификатор пресета, state_id — уникальный идентификатор состояния. Идентификаторы можно получить в подписке на пресеты, см. раздел А.7

Таблица А.8.2 — Структура ответа Reply на команду *SetPresetState*

Имя параметра	Тип параметра	Описание
request_answ	<i>RequestAnsw</i> { “request_id”: “clientrequestid”, “code”: http_status_code, “subscription_id”: null }	В поле <i>request_id</i> возвращается универсальный идентификатор “ someid ” из клиентского запроса. В поле <i>code</i> клиенту возвращается статус активации состояния пресета.
subscription_id	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
telemetry	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
params	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
location	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
trajectory	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся
presets	Не передаётся	В ответе на команду <i>SetParameter</i> это поле не передаётся

Пример запроса активации состояния пресета:

```
{  
  'method': 'SetPresetState',  
  'request_id': '67',  
  'preset': {'preset_id': 'weather2.ini', 'state_id': '1'}  
}
```

Пример ответа — удачная активация состояния пресета:

```
{  
  "request_answ": {  
    "request_id": "67",  
    "code": 200,  
    "subscription_id": null  
  },  
  "subscription_id": null  
}
```

Приложение Б (справочное)

Использование приложения LorisTool для просмотра имитированных траекторий

Б.1 Имитированные траектории

Б.1.1 Имитированная траектория добавляется в виджете *Map* через контекстное меню (правый клик в области карты, *Add imitated target*), рисунок Б.1, далее указать высоту (*Altitude*), нажать *Add*.

Б.1.2 Для отображения пиктограммы имитированной траектории в интерфейсе Программы (рисунок Б.2) убедиться, что в виджете *Configuration* (секция *Map display / Tracks / Classes*) включено отображение класса *Imitated target*. Информация о траектории («карточка цели») вызывается и скрывается двойным левым кликом по пиктограмме.

Б.1.3 Для получения имитированной траектории в API убедиться, что в виджете *Configuration* (секция *API / Classes*) включен класс *Imitated target*.

Б.1.4 Правым кликом по пиктограмме (рисунок Б.3) вызывается контекстное меню, с помощью которого можно удалить траекторию, удалить все имитированные траектории и включить (отключить) ручное сопровождение (поле *is_manual* API).

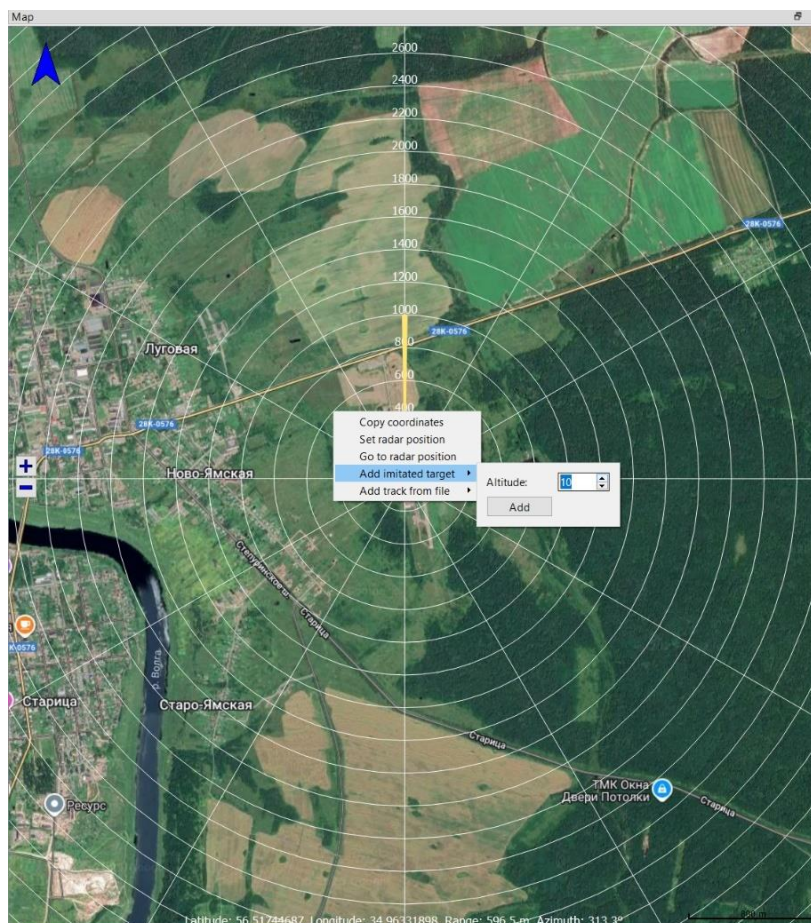


Рисунок Б.1 — Добавление имитированной траектории

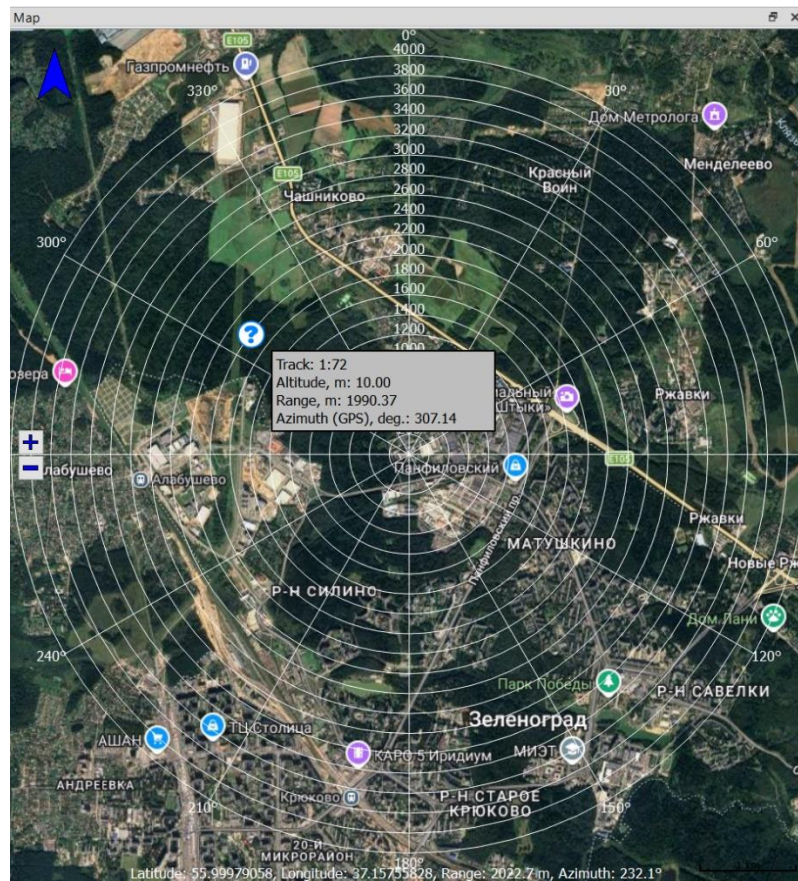


Рисунок Б.2 — Отображение имитированной траектории в интерфейсе Программы

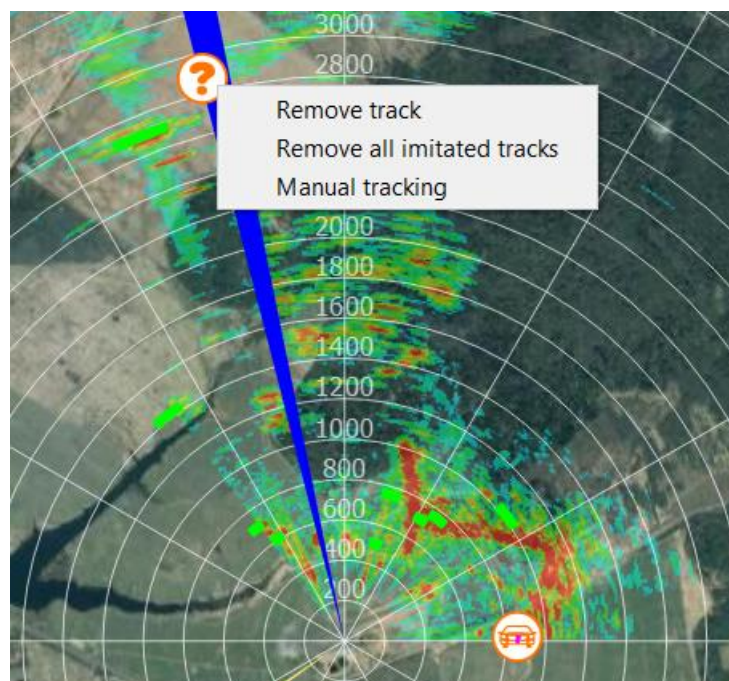


Рисунок Б.3 — Контекстное меню имитированной траектории

Б.2 Проигрывание траектории из файла

Б.2.1 Проигрывание траектории из файла включается в виджете *Map* через контекстное меню (правый клик в области карты, *Add track from file*), рисунок Б.4, нажать *Browse file*, выбрать файл (файлы).

Б.2.2 Для отображения пиктограммы траектории в интерфейсе Программы (рисунок Б.5) убедиться, что в виджете *Configuration* (секция *Map display / Tracks / Classes*) включено отображение соответствующего класса. Информация о траектории («карточка цели») вызывается и скрывается двойным левым кликом по пиктограмме.

Б.2.3 Для получения траектории в API убедиться, что в виджете *Configuration* (секция *API / Classes*) включен соответствующий класс.

Б.2.4 Правым кликом по пиктограмме вызывается контекстное меню, с помощью которого можно удалить траекторию, удалить все имитированные траектории и включить (отключить) ручное сопровождение (поле *is_manual* API).

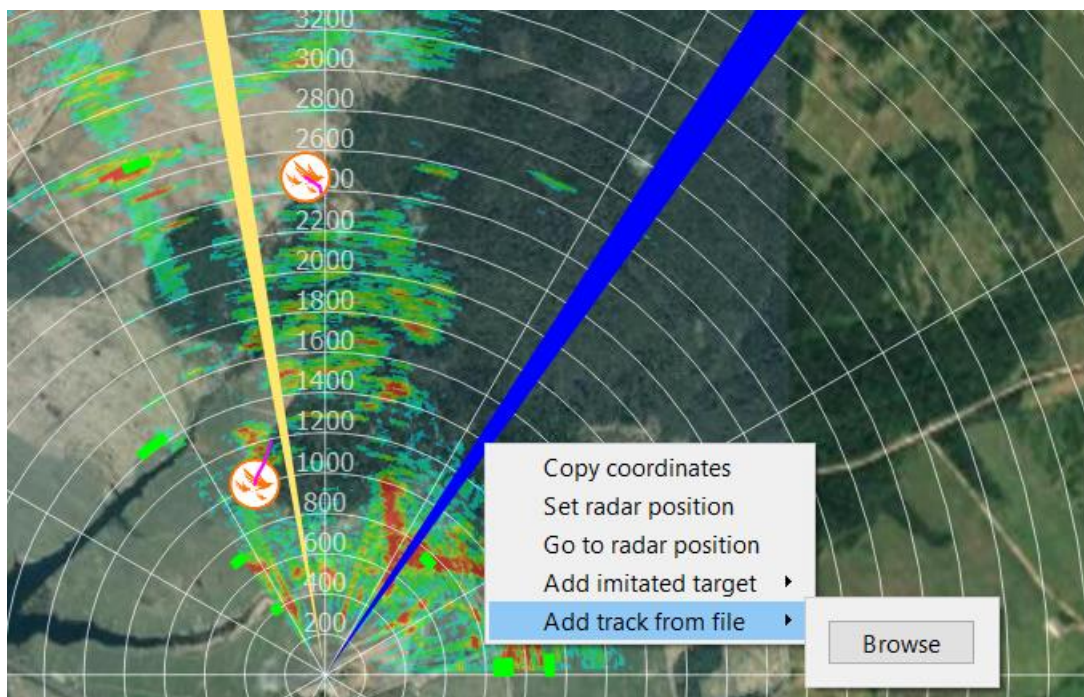


Рисунок Б.4 — Добавление траектории из файла (файлов)

Б.2.5 Файл траектории — текстовый, содержит строки с координатными точками траектории в местной системе координат, отсортированные по увеличению времени, характеристики в каждой строке указываются последовательно с разделителем табуляции.

Б.2.6 Запись файлов траекторий включается (отключается) в **Loris Tool** через виджет конфигурации (*Recording / Track writing*), файлы сохраняются в каталоге *apps* каталога с установленным приложением в папке с именем вида *2026-01-22-15-23-41_tracks*, либо в каталог, установленный в виджете конфигурации.

Б.3 Проигрывание файла сырых данных с помощью приложения **Bacon player**

Б.3.1 Технологическое приложение **Bacon player** из состава комплекта программного Loris Tool предназначено для проигрывания радиолокационных данных из файла.

Вид графического интерфейса приложения: рисунок Б.5.

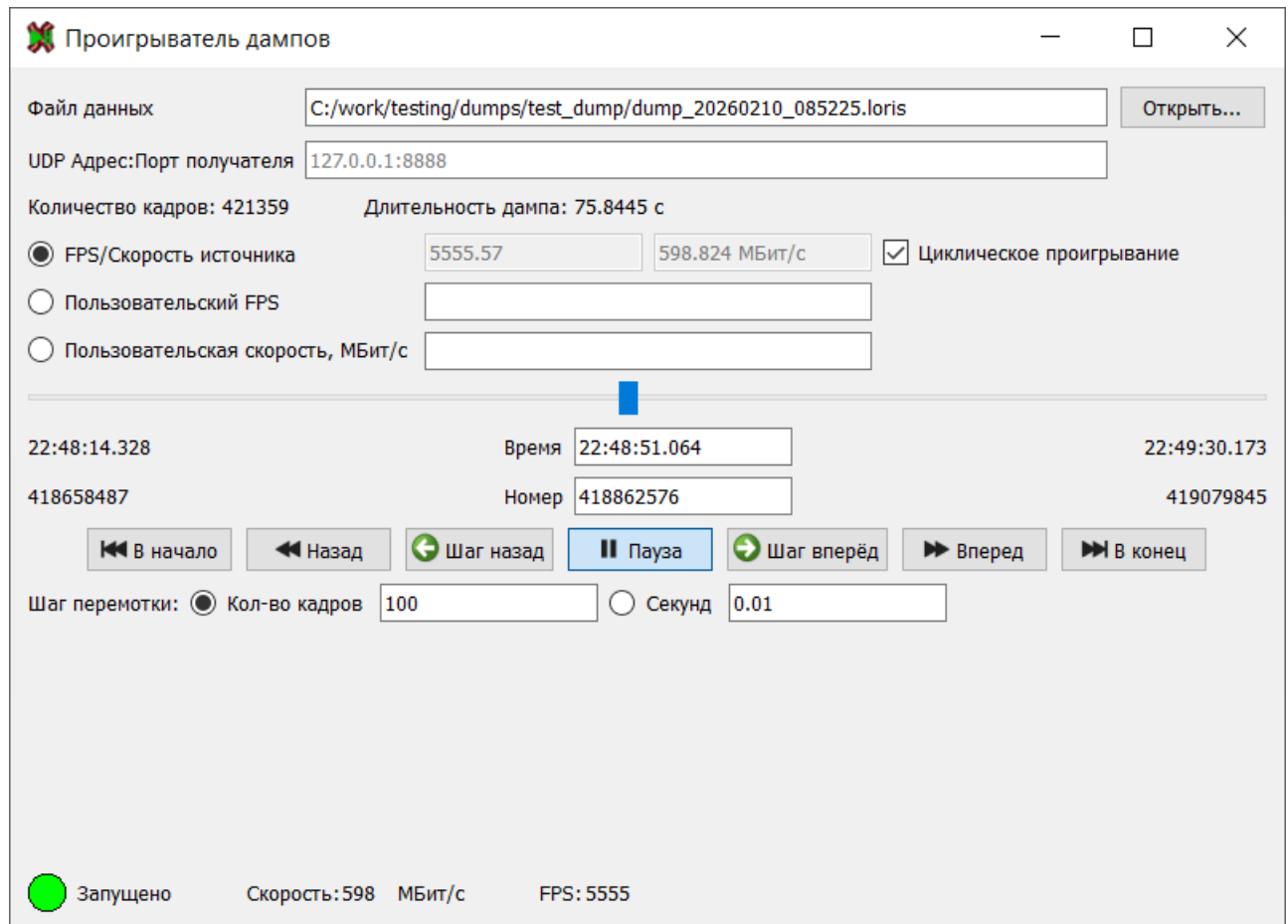


Рисунок Б.5 — Графический интерфейс приложения **Bacon player**

Б.3.2 Запись файла сырых данных включается (отключается) в **Loris Tool** через виджет конфигурации (*Recording / Dump writing*), файлы сохраняются в каталоге *dump* каталога с установленным приложением.

Приложение В (справочное)

Включение, проверка работоспособности и отключение РЛС с использованием API

В.1 Информация об использовании API для включения (отключения) и проверки работоспособности РЛС приведены в таблице В.1. Порядок действий оператора изложен в Руководстве оператора комплекта программного Loris Tool ЛУБА.00050-01 34 01.

Таблица В.1 — Включение, проверки работоспособности и отключение РЛС

Действие	Метод, поле	Тип	Ожидаемое значение	Примечание
Проверка состояния соединения по интерфейсу управления	TelemetrySubscription TelemetryData. control_connection_state	string	“ok” — соединение есть; “not_connected” — соединения нет	
Проверка состояния интерфейса к трансиверу	TelemetrySubscription TelemetryData. transceiver_interface_status	string	“ok” — ответы есть; “error” — ответов нет; “not_connected” — соединения нет	
Проверка отсутствия аварии трансивера	TelemetrySubscription TelemetryData. transceiver_failure	string	“ok” — норма; “error” — авария; “not_connected” — соединения нет	
Проверка статуса петли ФАПЧ	TelemetrySubscription TelemetryData. transceiver_pll_status	string	“ok” — захват есть; “error” — захвата нет; “not_connected” — соединения нет	
Проверка состояния интерфейса к контроллеру электропривода	TelemetrySubscription TelemetryData. rotator_status	string	“ok” — ответы есть; “error” — ответов нет; “not_connected” — соединения нет	
Включение (отключение) рабочего режима	SetParameter (установка) / ParametersSubscription (проверка) ParametersData.control_operating	bool	true (включить), false (отключить)	
Проверка включения рабочего режима	TelemetrySubscription TelemetryData. preproc_operating_state	string	“ok” — включено; “error” — отключено; “not_connected” — нет соединения	
Проверка наличия потока в интерфейсе передачи данных	TelemetrySubscription TelemetryData. data_stream_status	string	ok” — поток есть; “error” — потока нет	

Действие	Метод, поле	Тип	Ожидаемое значение	Примечание
Проверка переполнения очереди сигнальной обработки	TelemetrySubscription TelemetryData. processing_overflow	string	“ok” — норма; “warning” — предупреждение; “error” — переполнение; “not_connected” — нет потока данных	
Проверка потерь в интерфейсе передачи данных	TelemetrySubscription TelemetryData. lost_frames_percent	string	Пример: “0.00 %”	Допустимым считается не более 2 %
Включение (отключение) приемника	SetParameter (установка) / ParametersSubscription (проверка) ParametersData. transceiver_strobe_rx	bool	true (включить), false (отключить)	
Включение (отключение) передатчика	SetParameter (установка) / ParametersSubscription (проверка) ParametersData. transceiver_strobe_tx	bool	true (включить), false (отключить)	
Включение (отключение) усилителя 1	SetParameter (установка) / ParametersSubscription (проверка) ParametersData. transceiver_strobe_pa1	bool	true (включить), false (отключить)	
Включение (отключение) усилителя 2	SetParameter (установка) / ParametersSubscription (проверка) ParametersData. transceiver_strobe_pa2	bool	true (включить), false (отключить)	
Проверка наличия мощности передатчика	TelemetrySubscription TelemetryData.transceiver_power	double	При (ParametersData.transceiver_strobe_tx == true): от 0,9 до 1,3	
Проверка наличия мощности усилителя 1	TelemetrySubscription TelemetryData.pa1_power	double	При (ParametersData.transceiver_strobe_tx == true) и (ParametersData.transceiver_strobe_pa1 == true): от 0,9 до 1,3	

Действие	Метод, поле	Тип	Ожидаемое значение	Примечание
Проверка наличия мощности усилителя 2	TelemetrySubscription TelemetryData.pa2_power	double	При (ParametersData.transceiver_strobe_tx == true) и (ParametersData.transceiver_strobe_pa2 == true): от 0,9 до 1,3	
Включение вращения *	SetParameter (установка) / ParametersSubscription (проверка) ParametersData.rotator_azimuth_control	bool	true (включить), false (отключить)	
Установка скорости вращения *	SetParameter (установка) / ParametersSubscription (проверка) ParametersData.rotator_azimuth_speed	double	Рекомендуемое значение от 2400 до 3000	
* При автоконфигурации, помимо скорости вращения, следует устанавливать "rotator_azimuth_control": true. Если необходимо остановить вращение в процессе работы, то следует либо установить "rotator_azimuth_speed": 0, либо "rotator_azimuth_control": false				

[illegible]