Федеральное государственное бюджетное учреждение науки
Федеральный исследовательский центр «Институт общей физики им. А.М. Прохорова»
Научный центр волновых исследований (филиал)

На правах рукописи

ЗАВОЗИН ВЛАДИМИР АЛЕКСАНДРОВИЧ

МИКРОДЖОУЛЬНОЕ ЛАЗЕРНОЕ ДИСТАНЦИОННОЕ ЗОНДИРОВАНИЕ МНОГОСЛОЙНЫХ СРЕД

Диссертация
на соискание учёной степени
кандидата физико-математических наук
1.3.19. Лазерная физика

Научный руководитель:

д. ф.-м.н Першин Сергей Михайлович

Москва 2024

Введение

Актуальность работы

Лазерное дистанционное зондирование широко применяется для исследования атмосферы и гидросферы с помощью лидаров (транслитерация от англ. LiDAR, Light Detection And Ranging — обнаружение и определение дальности с помощью света)[1]. Лидары регистрируют упруго и не упруго рассеянные фотоны, по которым определяют физико-химические свойства удаленного объекта. Несмотря на бурное развитие лидарных технологий после изобретения лазера, они не получили широкого распространения на практике, что связано с опасностью повреждения органов зрения.

Следовательно, становится актуальной задача разработки безопасных для органов зрения принципиально новых методов лазерного дистанционного зондирования окружающей среды с плотностью энергии на объекте облучения сравнимой или меньше солнечной. В 1991 году С. М. Першин с соавторами [2] предложил новый статистический принцип зондирования, когда вероятность регистрации сигнала рассеяния от лазерного импульса много меньше единицы. Он экспериментально обосновал этот способ зондирования с использованием микроджоульных импульсов диодных и других лазеров и стробируемых квантовых счётчиков на основе однофотонных лавинных фотодиодов с усилением сигнала до 10^6. Так была открыта новая эра лазерного зондирования среды обитания без каких-либо ограничений. Важным преимуществом предложенной технологии является высокая эффективность с малым энергопотреблением и возможность миниатюризации лидаров, а также устойчивость к внешним источникам засветки и бликам обратно рассеянного зондирующего излучения. Именно эти преимущества микроджоульного лидара, разработанного в России обеспечили его включение в состав миссии “Mars Polar Lander” НАСА (США) [3] в ходе конкурсной борьбы.

Предложенный подход в современном мире используется для коммерческих систем навигации беспилотных автомобилей, других подвижных платформ и

роботов. Однако, коммерческие приборы работают только до первого рассеивающего объекта. Это ограничивает применение таких лидаров в сложных погодных условиях (туманы, дожди, дымовые завесы). Другим недостатком коммерческих приборов является наличие «мертвой зоны», то есть невозможность проводить дальнометрию на малых расстояниях от лидара (единицы и десятки метров). Поэтому актуальной задачей является разработка статистических лидаров для навигации в сложных погодных условиях и стыковки вплоть до касания (без мертвой зоны измерения) с повышением точности определения расстояния до сантиметров.

Другой сферой использования лидаров является исследования климата, атмосферы и океана [4, 5]. Здесь актуальной задачей является создание прибора с безопасным для глаз уровнем излучения. Такой лидар обеспечивает построение трехмерной карты распределения многослойных сред таких как облака, туманы, шлейфы, информация о которых необходима для моделирования процессов изменения глобального климата, измерения массы загрязняющих веществ и экологических рисков от пожаров лесов, извержения вулканов и др.

Цель и задачи работы

Разработка новых подходов лазерного дистанционного зондирования микроджоульными импульсами со стробируемыми детекторами в режиме счета фотонов с целью повышения разрешающей способности дальнометрии и повышения чувствительности измерения вариации концентрации аэрозолей при прогнозирования тектонических событий.

Для достижения сформулированной цели были поставлены и решены следующие задачи:

1. Разработать способ генерации лазерного наносекундного импульса с пикосекундной длительностью фронта на диодном лазере для повышения разрешающей способности безопасной для глаз лазерной дальнометрии;
2. Исследовать влияние параметров многослойных оптически плотных сред на

формирование эхо-сигнала лидара в режиме счета фотонов;

3. Исследовать лидаром динамику аэрозолей в районах с высокой тектонической активностью;

Научная новизна

1. Экспериментально обоснован метод генерации пикосекундной ступеньки на переднем фронте наносекундного оптического импульса, при накачке инжекционного диодного лазера током разряда ёмкости, коммутируемого биполярным транзистором в лавинном режиме.

2. Предложен и экспериментально обоснован новый подход лазерной дальнометрии, кратно, до сантиметров, повышающий точность измерений расстояний при использовании диодного лазера с пикосекундной ступенькой на переднем фронте наносекундного оптического импульса и лавинного фотодиода. Режимом работы лидара управляют в зависимости от уровня сигнала обратного рассеяния (отношения количества срабатываний фотоприёмника к заданному числу импульсов). Так в режиме «слабого сигнала», вероятность регистрации обратно рассеянных фотонов много меньше единицы. Для измерения расстояния необходимо зарегистрировать статистику распределения интервалов времени между стартом генерации лазерного импульса и срабатыванием приёмника и по положению максимума на гистограмме определить удалённость зондируемого объекта. Точность измерения расстояния определяют шириной импульса на полувысоте. Во втором режиме работы лидара (приближение «сильного сигнала») вероятность регистрации обратно рассеянных фотонов близка к единице. Расстояние определяют по положению первой доминирующей ячейки гистограммы, которую заполняют фотоотчёты от пикосекундной ступеньки на фронте импульса. При этом пространственное разрешение дальнометрии улучшается на порядок и более, вплоть до длительности ступеньки на фронте зондирующего импульса.

3. Впервые прямым измерением микроджоульным лидаром была исследована

многослойная структура тумана. Продемонстрировано, что основанный на регистрации баллистических фотонов цифровой принцип работы микроджоульного лидара позволяет обнаруживать объекты сквозь оптические плотные среды (рассеивающие и неоднородные среды, зеркальные поверхности и шлейфы дымовой завесы, и др.).

4. Впервые в ходе лазерного мониторинга динамики аэрозолей в закрытом тоннеле Баксанской Нейтринной Обсерватории (БНО) обнаружена модуляция лидарного эхо-сигнала тектоническими газами, эмиссия которых вызвана деформацией коры Земли приливными волнами.

5. Впервые прямым измерением оптической прозрачности атмосферы в закрытом объёме подземной лаборатории обнаружена индуцированная ионами генерация аэрозолей.

Теоретическая и практическая значимость

Генерация пикосекундной ступеньки на фронте наносекундного оптического импульса на диодном лазере ставит вопрос о физике лазерной генерации при лавинной коммутации тока в ключе питания лазера, для чего требуется разработать теоретическую модель подобных процессов. Достигнутое сокращение длительности фронта наносекундного импульса лазера открывает перспективы разработки новых цифровых лидаров с повышенной точностью дальнометрии и возможностью измерения скорости в том числе и на коротких расстояниях. Этот факт имеет ключевое значение при автоматизированном маневрировании и стыковке беспилотных платформ. Отметим, что точность дальнометрии в лидаре в статистическом режиме работы определяется длительностью ступеньки на фронте лазерного импульса, временем развития лавины в приёмном фотодиоде и разрешающей способностью время-цифрового преобразователя (англ. Time to Digital Converter, TDC). Предложенный подход, учитывающий особенность генерации пикосекундной ступеньки в диодном лазере, позволяет управлять точностью определения расстояния и минимальным периодом измерений в

зависимости от количества срабатываний стробируемого лавинного фотодиода. Адаптационное управление работой лидара являются критическими важным при стыковке платформ в космическом пространстве. Здесь поиск причала на большом удалении проводят в режиме «слабого сигнала» с длинным (100-500 м) стробом и, напротив, сближение и стыковку проводят в режиме «сильного сигнала» с сантиметровом разрешением расстояния и скорости сближения около 1 см/с. Особым преимуществом здесь является цифровой способ обработки рассеянных фотонов, который позволяет вести зондирование без «мёртвой зоны» вплоть до касания и устойчив к повреждению приёмной аппаратуры мощным обратным сигналом или бликом от поверхности объекта зондирования. Последний фактор допускает использование светоотражающих покрытий поверхности причала, которые кратно повышают значение отношения сигнал/шум и, соответственно, дальность обнаружения и зондирования. Несомненно, что предложенный подход работы лидара является важной альтернативой использования дорогостоящих пикосекундных лазеров в лидарах, а, значит, делает лидары более доступными для различных научных и технических приложений.

Продемонстрированный подход лазерного зондирования многослойных рассеивающих сред (туманы, дымовые завесы, кроны деревьев, маскирующие сетки, зеркальные поверхности) позволяет беспилотникам маневрировать в сложных погодных условиях. Использование лидара не ограничивается атмосферными измерениями, но также позволяет определять слои-преграды и препятствия под водой. Статистический режим работы лидара позволяет получать трёхмерные карты распределения физических параметров многослойных сред таких как облака или плотные туманы и шлейфы дыма, что имеет большое значение для построение надёжных климатических моделей и оценки степени загрязнения атмосферы продуктами горения.

В работе проведён длительный эксперимент по мониторингу динамики вариации аэрозолей в закрытых объёмах в сейсмически активных регионах с целью поисков новых индикаторов тектонической активности. Обнаруженная модуляция

лидарного эхо-сигнала тектоническими газами, эмиссия которых вызвана приливными волнами, открывает новый канал мониторинга деформации земной коры, что важно для построения надежных теоретических моделей геодинамики Земли. В частности, тектонические события сопровождаются повышенной эманацией радиоактивного газа радона [6]. Поэтому дополнительным индикатором при мониторинге служит факт генерации аэрозолей отрицательными ионами, индуцируемых альфа-частицами радона.

Методология и методы исследования

1. Метод инжекционной накачки был применен для генерации пикосекундной ступеньки на фронте наносекундного импульса диодного лазера
2. Принцип работы разработанного безопасного для глаз лидара базируется на методологии время-пролетной дальнометрии в режиме счета фотонов и время-цифрового преобразователя
3. Количественные измерения растяжения-сжатия коры Земли были измерены методом лазерной интерферометрии для мониторинга тектонических процессов

Положения, выносимые на защиту

1. Накачка инжекционного диодного лазера током разряда емкости, коммутируемой биполярным транзистором в лавинном режиме работы сопровождается генерацией пикосекундной ступеньки на фронте оптического импульса

2. Диодный лазер с пикосекундной ступенькой на фронте наносекундного оптического импульса в лидаре на стробируемом лавинном фотодиоде в режиме счета фотонов обеспечивает сантиметровую точность дальнометрии

3. Лидар на однофотонном приёмнике обеспечивает зондирование многослойных рассеивающих сред, несмотря на потери в каждом слое-преграде с

высокой оптической плотностью, с вероятностью обратного рассеяния близкой к единице.

4. Тектонические процессы, во время которых повышается концентрация аэроионов, индуцируемых альфа-частицами радона, приводит к образованию аэрозолей и понижению коэффициента оптического пропускания трассы с коэффициентом корреляции $\rho=0.89$.

Степень достоверности результатов

Достоверность результатов, полученных в диссертации, подтверждается использованием для исследований современного высокоточного оборудования, их воспроизводимостью, а также докладами на международных и национальных конференциях и публикациями в ведущих отечественных и мировых научных журналах.

Личный вклад диссертанта

Планирование и проведение измерений, обработка экспериментальных данных и интерпретация полученных результатов выполнены автором лично либо при его непосредственном участии. Написание и обсуждение текстов статей и тезисов конференций выполнено в соавторстве при непосредственном участии автора.

Апробация результатов работы

Основные результаты диссертационной работы были представлены и обсуждены на семинарах Института общей физики им. А.М. Прохорова РАН (Москва, 19 февраля 2020 г., 20 октября 2021 г., 14 сентября 2022 г., 29 марта 2023 г.), а также на национальных и международных конференциях:

— The International Conference on Advanced Optoelectronics and Lasers, CAOL 2019;

— The International Conference Laser Optics 2020, ICLO 2020;

— Школа-конференция молодых ученых «ПРОХОРОВСКИЕ НЕДЕЛИ», ИОФ РАН 2021;
— V International Conference on Ultrafast Optical Science «UltrafastLight-2021»;
— International Conference on Advanced Laser Technologies (ALT) 2022;
— The International Conference Laser Optics 2022 (ICLO 2022).

Работа была поддержана грантом Российского научного фонда № 19-19-00712 период выполнения 2019-2022 с продлением № 19-19-00712-П на период 2022-2023.

Публикации

Результаты диссертационной работы опубликованы в 14 печатных работах в рецензируемых научных журналах из списка ВАК и в 11 тезисах конференций.

Тезисы конференций

1. S. M. Pershin, G. I. Dolgikh, V. S. Makarov, A. V. Turin, M. Y. Grishin, **V. A. Zavozin**, V. N. Lednev, A. A. Plotnikov. A pulsed diode laser for tectonic aerosol lidar sensing // Proceedings of the International Conference on Advanced Optoelectronics and Lasers, CAOL. – 2019. – Vols. 2019-Septe.
2. М. Я. Гришин, **В. А. Завозин**. Лидарное зондирование тектонического аэрозоля как новый метод мониторинга деформаций коры Земли / М. Я. Гришин, В. А. Завозин // Тезисы докладов Школы-конференции молодых ученых «Прохоровские недели». – Федеральное государственное бюджетное учреждение науки Федеральный …, 2020. – С. 6-8.
3. S. M. Pershin, M. Ya. Grishin, **V. A. Zavozin**, V. N. Lednev, V. S. Makarov, P. A. Sdvizhenskii, A. V. Turin. Eye-safe LIDAR sensing through dense fog // Proceedings - International Conference Laser Optics 2020, ICLO 2020. – 2020.
4. S. M. Pershin, A. L. Sobisevich, M. Y. Grishin, **V. A. Zavozin**, V. V. Kuzminov, V. N. Lednev, D. V. Likhodeev. Tectonic aerosol sensing by lidar as a new technique for Earth’s crust deformation monitoring // Proceedings - International Conference Laser

Optics 2020, ICLO 2020 / Citation Key: Pershin2020. – Institute of Electrical and Electronics Engineers Inc., 2020.

5. **В. А. Завозин**, М. Я. Гришин. Лазерное зондирование многослойных туманов лидаром с безопасным для глаз уровнем излучения / В. А. Завозин, М. Я. Гришин // Тезисы докладов Школы-конференции молодых ученых «Прохоровские недели». – Федеральное государственное бюджетное учреждение науки Федеральный …, 2021. – С. 10-12.

6. S. M. Pershin, M. Ya. Grishin, **V. A. Zavozin**, V. S. Makarov, V. N. Lednev, A. V. Myasnikov, A. V. Turin. A 3-ns pulsed diode laser for a high spatial resolution lidar // Abstracts of the 28th International Conference on Advanced Laser Technologies. – 2021. – P. 171.

7. S. M. Pershin, M. Ya. Grishin, **V. A. Zavozin**, V. S. Makarov, V. N. Lednev, A. V. Myasnikov. Omnidirectional dynamics of the Earth's crust seasonal deformation and the aerosol output decrease in the adit over the Elbrus volcano magmatic chamber // Abstracts of the 28th International Conference on Advanced Laser Technologies. – 2021. – P. 172.

8. S. M. Pershin, V. S. Makarov, M. Ya. Grishin, **V. A. Zavozin**, I. M. Tupitsyn, E. A. Cheshev, D. G. Artemova. Unique dual-pulse diode laser with controllable (3-20 ns) interval & duration for the eye-safe lidar // V International Conference on Ultrafast Optical Science "UltrafastLight-2021". – Moscow, 2021. – P. 179.

9. V. Lednev, M. Y. Grishin, P. Sdvizhenskii, **V. Zavozin**, S. Pershin, A. Bunkin. Developing Compact LIDAR Systems: Size and Eye-Safety Matters // Сборник трудов конференции "International Conference on Advanced Laser Technologies (ALT)". – Федеральное государственное бюджетное учреждение науки Федеральный …, 2022. – P. 194-194.

10. S. M. Pershin, **V. A. Zavozin**, D. G. Artemova, M. Ya. Grishin, K. Kh. Kanonidi, V. N. Lednev, V. S. Makarov, Ya. Ya. Ponurovskii. Improving magmatic aerosol detection by LIDAR above the Elbrus magma chamber // 2022 International Conference Laser Optics (ICLO) / Citation Key: 9839805. – 2022. – P. 1.

11. S. M. Pershin, **V. A. Zavozin**, M. Y. Grishin, V. N. Lednev, G. A. Boldin, L. B. Bezrukov, A. K. Mezhokh, V. V. Sinev. Air ions induced aerosol sensing by eye-safe LIDAR. – 2023.

Научные публикации

1. С. М. Першин, Г. И. Долгих, А. Ф. Бункин, М. Я. Гришин, **В. А. Завозин**, В. К. Клинков, В. Н. Леднёв, В. С. Макаров, А. А. Плотников, А. В. Тюрин. Корреляции сигналов лидарного аэрозольного рассеяния и лазерного деформографа при сжатии/растяжении коры Земли // Краткие сообщения по физике ФИАН. – 2018. – Т. 45. – № 7. – С. 32-38. – DOI: 10.3103/S1068335618070059.
2. С. М. Першин, А. Н. Федоров, А. В. Тюрин, А. В. Мясников, В. С. Макаров, В. Н. Леднёв, В. В. Кузьминов, **В. А. Завозин**, М. Я. Гришин, В. Б. Петков. Лидарное зондирование эволюции многослойных туманов в наклонном тоннеле Баксанской нейтринной обсерватории // Краткие сообщения по физике ФИАН. – 2019. – Т. 10. – С. 46-54.
3. S. M. Pershin, A. L. Sobisevich, M. Y. Grishin, V. V. Gravirov, **V. A. Zavozin**, V. V. Kuzminov, V. N. Lednev, D. V. Likhodeev, V. S. Makarov, A. V. Myasnikov, A. N. Fedorov. Volcanic activity monitoring by unique LIDAR based on a diode laser // Laser Physics Letters. – 2020. – Vol. 17. – № 11. – P. 115607. – DOI: 10.1088/1612-202X/abbedc.
4. S. M. Pershin, M. Y. Grishin, **V. A. Zavozin**, V. N. Lednev, V. A. Lukyanchenko, V. S. Makarov. Aerosol layers sensing by an eye-safe lidar near the Elbrus summit // Laser Physics Letters. – 2020. – Vol. 17. – № 2. – P. 026003. – DOI: 10.1088/1612-202X/ab66c4.
5. С. М. Першин, М. Я. Гришин, **В. А. Завозин**, В. Н. Леднев, А. Н. Федоров, А. В. Мясников, А. В. Тюрин. Диодный лазер, генерирующий импульсы 3 нс, для лидара с высоким пространственным разрешением // Квант. электроника. – 2021. – Т. 51. – № 5

6. С. М. Першин, А. Л. Собисевич, М. Я. Гришин, **В. А. Завозин**, В. С. Макаров, В. Н. Леднёв, А. Н. Фёдоров, А. В. Мясников, Д. Г. Артёмова. Разнонаправленная модуляция сезонного сжатия коры Земли и сигнала аэрозольного лидара в тоннеле над очагом вулкана Эльбрус // Доклады Российской академии наук. Физика, технические науки. – 2021. – Т. 501. – № 1. – С. 14-18. – DOI: 10.31857/S2686740021060134.

7. **V. A. Zavozin**, M. Y. Grishin, V. N. Lednev, V. S. Makarov, S. M. Pershin. Eye-safe photon counting LIDAR for magmatic aerosol detection // Laser Physics. – 2022. – Vol. 32. – № 12. – P. 125601. – DOI: 10.1088/1555-6611/aca15d.

8. V. Myasnikov, S. M. Pershin, M. Ya. Grishin, **V. A. Zavozin**, V. S. Makarov, A. A. Ushakov. Estimation of the Influence of Meteorological Factors on the Aerosol Lidar Signal in Tunnels above the Elbrus Volcano Chamber // Physics of Wave Phenomena. – 2022. – Vol. 30. – № 2. – P. 119-127. – DOI: 10.3103/S1541308X22020054.

9. С. М. Першин, В. С. Макаров, М. Я. Гришин, **В. А. Завозин**, А. Л. Коромыслов, В. Н. Леднев, П. А. Сдвиженский, И. Прохазка, И. М. Тупицын, Е. А. Чешев. Новый режим генерации диодного лазера: 200-пикосекундный фронт наносекундного импульса // Квантовая электроника. – 2022. – Т. 52. – № 11. – С. 1050-1053.

10. С. М. Першин, А. Л. Собисевич, **В. А. Завозин**, М. Я. Гришин, В. Н. Леднев, В. Б. Петков, Я. Я. Понуровский, А. Н. Фёдоров, Д. Г. Артёмова. Лидарное детектирование аэрозолей в тоннеле над очагом вулкана Эльбрус // Краткие сообщения по физике ФИАН. – 2022. – Т. 49. – № 2. – С. 10-19.

11. С. Першин, Е. Гордеев, М. Гришин, **В. Завозин**, В. Макаров, В. Леднёв, Я. Понуровский, А. Фёдоров, А. Ушаков, В. Казалов. Реверсия конвекции воздуха в горячем тоннеле над очагом вулкана Эльбрус // Краткие сообщения по физике Физического института им. П.Н. Лебедева Российской Академии Наук. – 2023. – Т. 50. – № 3. – С. 3-13.

12. С. М. Першин, **В. А. Завозин**, В. Н. Леднев, Г. А. Болдин, М. Я. Гришин, В. С. Макаров, Л. Б. Безруков, А. К. Межох, В. В. Синев. Лидарный мониторинг

динамики аэрозолей, индуцированных аэроионами // Краткие сообщения по физике ФИАН. – 2023. – № 12. – С. 69.

13. С. М. Першин, А. Л. Собисевич, В. С. Макаров, А. В. Мясников, М. Я. Гришин, **В. А. Завозин**, В. Н. Леднев, Д. В. Лиходеев, В. В. Казалов. Лидарный мониторинг магматической активности малой камеры эльбрусского вулканического центра // Доклады Российской академии наук. Физика, технические науки. – 2023. – Т. 509. – № 1. – С. 15-20. – DOI: 10.31857/S2686740023020086.

14. S. M. Pershin, E. I. Gordeev, M. Ya. Grishin, **V. A. Zavozin**, V. S. Makarov, V. N. Lednev, Ya. Ya. Ponurovskii, D. B. Stavrovskii, A. A. Ushakov, V. V. Kazalov. Monitoring the Baric Modulation of Gas Concentration in the Baksan Neutrino Observatory Tunnel in the Elbrus Region Using Differential Absorption Lidar // Doklady Earth Sciences. – 2024. – Vol. 515. – № 1. – P. 535-540. – DOI: 10.1134/S1028334X23603164.

Благодарности.

Автор диссертации благодарен за активное участие в выполнении работы, консультации и обсуждения д.ф.-м.н. С.М. Першину, к.ф.-м.н. В.Н. Ледневу, к.ф.-м. н. М.Я. Гришину, сотруднику ИКИ РАН В.С. Макарову, к.ф.-м.н. А. В. Мяскникову, сотрудникам Баксанской Нейтринной Обсерватории С.П. Якименко, В.В. Казалову, сотруднику ИВиС ДВО РАН Р.Р. Акбашеву, д.ф.-м.н. Л.Б. Безрукову, сотрудникам центра биофотоники ИОФ РАН. Автор благодарен за поддержку исследований РНФ грант 19-19-00712.

Структура и объем работы.

Диссертация состоит из введения, 5 глав, заключения, списка цитируемой литературы и приложений. Общий объём диссертации составляет 121 страницу, включая 27 рисунков и 3 таблицы. Список цитируемой литературы содержит 100 наименований.

Глава 1. Обзор литературы

1.1 Безопасное для глаз лазерное дистанционное зондирование

Лазерное дистанционное зондирование широко применяется для исследования атмосферы и гидросферы с помощью лидаров (транслитерация от англ. LiDAR, Light Detection And Ranging — обнаружение и определение дальности с помощью света)[1]. Лидары регистрируют упруго и не упруго рассеянные фотоны, по которым определяют физико-химические свойства исследуемого объекта. Одним из основных направлений использования лидарной техники является исследование атмосферы и гидросферы. С помощью лидарного зондирования анализируют такие характеристики, как концентрация различных газов, аэрозолей [7, 8, 9, 10], скорость и направление ветра [11], температуру и влажность воздуха, и другие параметры атмосферы [12], а также состояние водной поверхности [13].

Однако, основным ограничением широкого применения лидаров остаётся опасность поражения органов зрения обитателей Земли. Следовательно, становится актуальной задача разработки безопасных для органов зрения принципиально новых методов лазерного дистанционного зондирования окружающей среды с плотностью энергии на объекте облучения сравнимой или меньше солнечной. В 1991 году С. М. Першин с соавторами [2]. предложил новый статистический принцип зондирования, когда вероятность регистрации сигнала рассеяния от лазерного импульса много меньше единицы. Он экспериментально обосновал этот способ зондирования с использованием микроджоульных импульсов диодных и других лазеров и квантовых счетчиков на основе однофотонных лавинных фотодиодов. Так была открыта новая эра лазерного зондирования среды обитания без каких-либо ограничений.

Применение новой методики зондирования значительно усовершенствовало работу с лидарными системами в повседневных условиях, гарантируя безопасное для зрения взаимодействие с окружающей средой. Этот подход базируется на

особенностях статистического режима зондирования лидаром, при котором мала вероятность (менее 1 %) уловить отклик от лазерного импульса при его взаимодействии с окружающей средой. Для повышения вероятности улавливания отклика увеличивается количество зондирующих импульсов на одно измерение до нескольких сотен тысяч. Доступность такого метода обеспечивается за счет применения диодных лазеров с небольшими габаритами, способных генерировать высокочастотные серии импульсов при уровне излучения, безопасном для глаз. Важную роль играет также цифровая обработка сигналов, благодаря чему метод демонстрирует высокую устойчивость к различным помехам. На основе этого принципа был разработан лидар на диодном лазере с принудительным запуском и энергией импульса 0.5 мкДж, что обеспечивало безопасность для глаз. В 1996 году [14] этот лидар рекордно малой массы, 940г, и энергопотребления, 0.2 Вт, был выдвинут на международный отборочный конкурс экспертов НАСА (США) и был включён в миссию “MARS POLAR LANDER” НАСА на Марсе для зондирования динамики атмосферы и пылевых бурь [3].

Рассмотрим принцип формирования лидарного сигнала обратного рассеяния в цифровом режиме. Стартовый импульс подаётся на лазер для запуска генерации излучения одновременно с устройством для счета времени. Далее лазерный импульс направляется на объект исследования. Для регистрации рассеянного излучения на лавинный однофотонный фотодиод подаётся пороговое напряжения. Когда на приёмник попадает излучение достаточной энергии для развития лавины, то фронт этого импульса является сигналом для остановки измерения времени (Рисунок 1а). В современных лидарах используется временно-цифровой преобразователь (англ. Time to Digital Converter, TDC) для регистрации времени задержки между стартовым импульсом и сигналом с однофотонного лавинного фотодиода. Многократное измерение времени задержки позволяет собрать гистограмму распределения времени срабатывания фотодиода по интервалам (Рисунок 1б). Важно отметить, что вероятность регистрации рассеянного излучения должна быть много меньше единицы, иначе гистограмма распределения

времени срабатывания фотодиода по интервалам будет представлять собой одну ячейку.

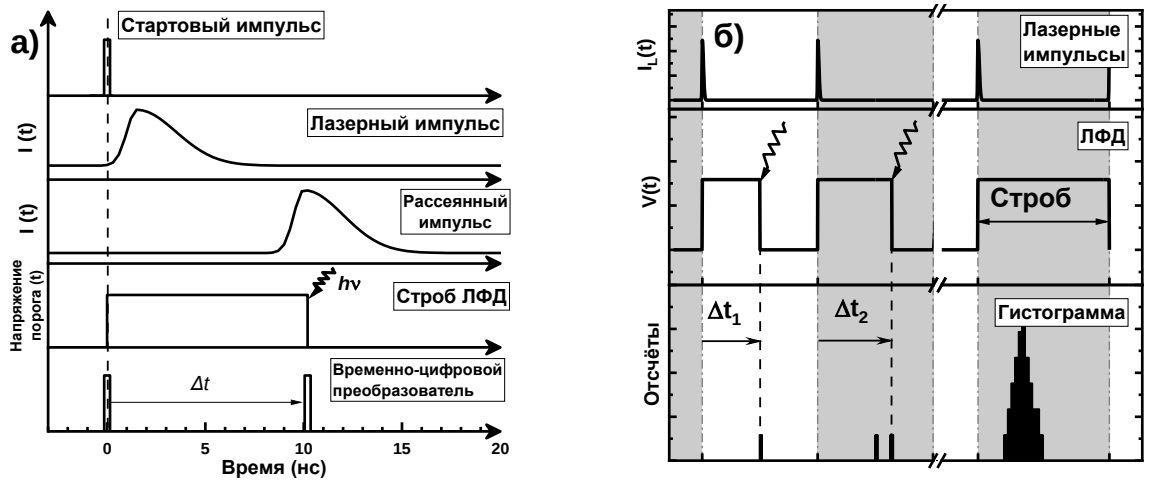

Рисунок 1 Принцип работы лидара в статистическом режиме. (а) - единичный акт измерения времени пролёта фотона до препятствия и обратно. (б) – формирование гистограммы фотоотсчётов при многократном измерении времени пролёта.

Для определения энергии на приёмной площадке лавинного фотодиода воспользуемся основным лидарном уравнением [7] для рассеиваемой энергии лазера, регистрируемой за время отклика детектора τ_d.

$$E(\lambda, R) = E_L \xi(\lambda) T(R) \xi(R) \frac{A_0}{R^2} \beta(\lambda_L, \lambda, R) \frac{c\tau_d}{2} \qquad (1.1)$$

где E_L – выходная энергия лазерного импульса, $\xi(\lambda)$ – коэффициент спектрального пропускания приемной оптической системы, $T(R)$ – коэффициент пропускания, $\xi(R)$ – геометрический фактор, $\beta(\lambda_L, \lambda, R)$ – объёмный коэффициент обратного рассеяния, $\frac{A_0}{R^2}$ – телесный угол, в котором осуществляется прием сигналов оптической системой (A_0 – площадь линзы или зеркала объектива), λ_L – длина волны лазера, λ – длина волны регистрируемого излучения, R – расстояние от лидара. В случае регистрации упругого обратного рассеяния длина волны регистрируемого излучения будет совпадать с лазерным. В цифровом лидаре для регистрации обратно рассеянного излучения используется однофотонный лавинный фотодиод, для возбуждения которого достаточно энергии сопоставимой с единичным фотоном.

Важной характеристикой счётчика фотонов является время восстановления до рабочего состояния. Восстановление напряжения на лавинном однофотонном фотодиоде до напряжения питания занимает от нескольких сотен наносекунд до нескольких микросекунд. Это является ограничением для быстрого сбора статистики в лидаре в цифровом режиме работы. За время прохождения лазерного импульса по трассе зондирования фотодиод срабатывает один раз. Поэтому статистика распределения времени срабатывания приёмника по интервалам будет искажена, поскольку вероятность зарегистрировать обратно рассеянный фотон будет подчинена распределению Паскаля [15]. Однако, если ввести допущения, что за время регистрации статистики свойства атмосферы, поверхности мишени и других препятствий на пути следования лазерного импульса меняются незначительно, что соответствует приближению стационарного потока независимых случайных событий, то можно сделать переход в вероятностное пространство идеального приёмника [15].

Обозначим k – количество интервалов времени на который разделён строб T. Тогда пространство вероятностных событий Ω_1 (для приёмника с большим временем восстановления) и Ω_2 (для идеального приёмника без времени на восстановление) построено следующим образом:

$$t_в = 0;\ \ \Omega_1 = P_i;\ \ i = 1, \dots, k;\ \ T < t_в < T_0;\ \ \Omega_2 = \theta_i;\ \ i = 1, \dots, k;$$
$$P_i; t \in [(i-1)\tau; i\tau]; \theta_i = \prod_{j=1}^{i-1}\left(1 - \tilde{P}_j\right)\tilde{P}_i;\ t \in [(i-1)\tau; i\tau]. \qquad (1.2)$$

где $t_в$ – время восстановления работы приёмника, T – время строба, T_0 – период включения строба, P_i – распределение вероятностей по интервалам для приёмника без времени восстановления, θ_i – распределение вероятностей по интервалам для приёмника с большим временем восстановления, t – время зондирования, τ – шаг дискретизации строба, $\tilde{P}_i$ – распределение вероятностей по интервалам для приёмника с временем восстановления.

Эти вероятностные пространства будут связаны следующим соотношением, при условии допущения, указанного ранее:

$$\theta = P_i exp\big(-\mu(1; i-1)\big) \qquad (1.3)$$

Это соотношение позволяет учитывать обратно рассеянное излучение, которое попадало на приёмную площадку за время строба, но не было зарегистрировано в следствии восстановления однофотонного лавинного фотодиода до рабочего состояния.

Несомненно, разработка цифрового режима работы лидаров нового поколения с безопасным для глаз уровнем излучения открыла новые направления в лазерном зондировании. Стали появляться разработки, направленные на уменьшение лазерного излучения в приборах, для возможности использования их с целью зондирования среды обитания без вреда здоровью человека. Самым известным применением лидаров нового поколения, спустя декаду, являются системы навигации беспилотных автомобилей и других подвижных платформ, использующих лидары кругового обзора [16, 17, 18].

Новая технология стала востребована в сканирующих лидарах. Первые разработки сканирующих систем с использованием лавинных фотодиодов была представлена в 2001 г Хайнриксом [19]. Позже в 2005 году в MIT Lincoln Laboratory был представлен лидар, на котором была установлена матрица 32x32 однофотонных фотодиодов-пикселей размером 20x20 микрон, работающих в режиме счета фотонов независимо друг от друга. В качестве лазера используется Nd:АИГ с пассивной модуляцией добротности и удвоителем частоты, лазерный импульс длительностью 300 пс, частота повторения 16 кГц [20].

В 2007 году компанией Velodyne был представлен первый сканирующий на 360 градусов лидар. В нем было расположено 64 лазера и 64 однофотонных лавинных фотодиода по вертикали, и вся эта конструкция вращалась со скоростью несколько (45-50) оборотов в секунду, что позволяло получать информацию об объектах вокруг прибора. В этом лидаре использовался диодный лазер с длинной волны 905 нм и малой энергией в импульсе, что позволяет его использовать, без вреда для органов зрения человека [21]. Так как полость внутри сетчатки глаза, которая расположена позади хрусталика, заполнена стекловидным телом

прозрачным для света с длиной волны 850 и 905 нм. В конце 2017 года компания представила лидар со 128 каналами излучения и приёмника [22]. А также была разработана компактная версия с 16 лазерами [23].

В таблице 1 перечислены существующие лидарные системы сканирующего типа, пригодные для использования в городских условиях [24].

Таблица 1. Характеристики коммерческих сканирующих лидаров

	Количество лазеров, шт	Расстояние детектирования, м	Длинна волны, нм	Угол обзора, град	Точность определения расстояния, см
Velodyne Model HDL-64S2	64	50	905	360	2
Velodyne Model HDL-64S3D	64	120	905	360	2
Ibeo LUX	4	50-200	905	85	4
Ibeo LUX HD	4	30-120	905	85	4
Ouster OS2	64	0.8-240	850	360	1.2

1.2 Импульсные полупроводниковые лазеры

Полупроводниковые лазеры, работающие в импульсном режиме, получили широкое распространение в науке и технике. В частности, они используются для накачки твердотельных лазеров [25]. Одно из главных преимуществ диодных лазеров это возможность модулировать излучение с помощью импульсов тока накачки [26]. Этот момент очень важен для практического применения, поскольку

на момент создания полупроводниковых гетероструктур уже существовали генераторы наносекундных импульсов на одном лавинном транзисторе [27]. Прямая модуляция полупроводниковых лазеров импульсами тока позволяла получать оптическую длительность излучения сопоставимую с длительностью приложенного тока [28]. Для накачки полупроводниковой структуры требуются импульсы тока от 2 до 10 А. Эти параметры обеспечиваются германиевыми низковольтными транзисторами со временем нарастания менее 1 нс. Однако, начальные разработки гетероструктур не позволяли изготавливать лазеры с большой энергией [29]. В дальнейшем для таких полупроводниковых лазеров потребовались большие токи накачки порядка 50-70 А, а для специальных приложений и до 200 А [27]. В 1990-х годах для этих целей стали использовать генераторы импульсов тока на лавинных биполярных транзисторах фирмы Zetex. Драйверы на этих транзисторах позволяли формировать токовый импульс длительностью 1.5 нс с амплитудой порядка 25А [27].

Также для формирования коротких импульсов на полупроводниковых лазерах известен метод под названием управление усилением (gain switching)[28]. Это метод, при котором оптическое усиление быстро переключается с низкого значения на высокое. Таким образом, полупроводниковый лазер, находящийся включённым ниже порога генерации, внезапно оказывается выше порога. После начальной задержки включения оптическое поле быстро нарастает в течение первого цикла релаксационных осцилляций и вынуждает лазер работать ниже порога за счёт процесса насыщения. Если длительность импульса тока выбрана таким образом, что ток отключается до формирования второго пика релаксационных осцилляций, то выходной сигнал состоит из короткого оптического импульса длительностью 10-20 пс, даже если импульс тока длится дольше 100 пс. Путём повторного наложения импульса тока можно генерировать последовательность пикосекундных импульсов с повторяемостью до нескольких гигагерц. Метод управления усилением был активно исследован в 1980-х годах. Наиболее полное теоретическое рассмотрение этого метода представлено в работе

1988 года[30]. С использованием этого метода были сгенерированы оптические импульсы длительностью 15 пс в 1981 [28], а позже в 1989 году удалось уменьшить до 6.7 пс [31], основываясь на теории описанной годом ранее [30]. До 1990 года удалось уменьшить длительность оптического импульса с 15 пс до 4 пс [32, 33, 34, 35]. В дальнейших работах не удалось снизить длительность лазерного импульса. Они были направлены на оптимизацию различных параметров, чтобы повысить коэффициент полезного действия таких лазеров [36, 37, 38].

Диодные лазеры с управляемым усилением потенциально могут быть идеальными источниками оптических импульсов для времяпролетных лидаров из-за их надёжности, удобства в использовании, долговечности, оптической мощности и высокой частоты генерации импульсов. Более того, они могут генерировать оптические импульсы с гибкими повторяемыми частотами от одиночного до частот выше гигагерц. Длительность оптических импульсов обычно принимает значения порядка нескольких десятков пикосекунд, что позволяет улучшить пространственное разрешение лидара в статистическом режиме. Отметим, что минимально возможная длительность генерации ультракоротких оптических импульсов до сих пор не оценена, поскольку ещё не до конца понята физика процессов переключения усиления полупроводникового лазера [37].

Возвращаясь к применению таких лазеров в лидарах, в частности времяпролетных, в 1997 году была продемонстрирована возможность генерировать импульсы длительностью порядка 10 нс на полувысоте [39]. В данной работе использовался генератор импульсов тока на основе транзистора в лавинном режиме переключения и были протестированы схемы накачки с использованием разных транзисторов и диодных лазеров. При оптимизации остальных параметров удалось добиться длительности оптического импульса 7.2 нс мощностью 72 Вт. Однако на осциллограммах наблюдается короткий фронт с быстрым нарастанием порядка нескольких десятков пикосекунд (Рисунок 2).

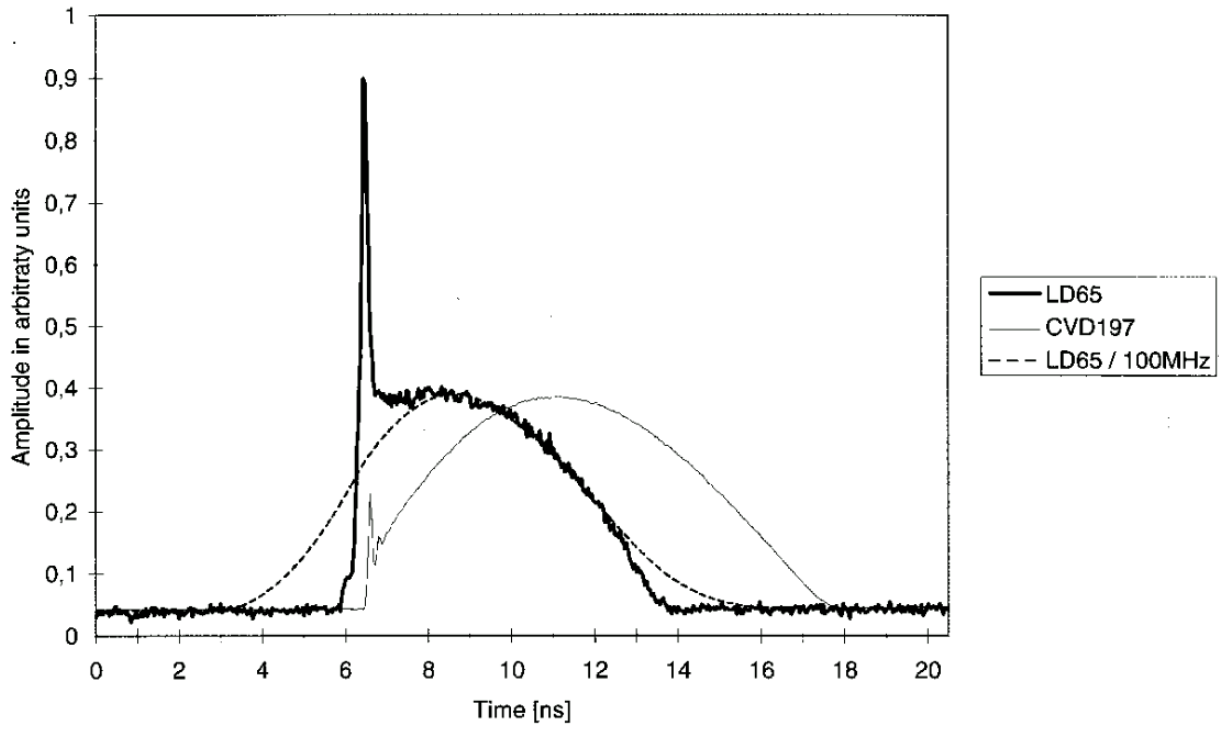

Рисунок 2 Оптические импульсы лазеров LD65 и CVD197 с полной полосой пропускания системы считывания и дополнительно с полосой 100 МГц для LD65 [39].

Это явление связано с переключением усиления. Импульс тока при лавинном режиме работы имеет неравномерное нарастание и при изменении скорости носителей заряда через лазерный диод происходит резкое падение скорости генерации как при управлении усилением описанной в работе [30]. В свою очередь неравномерное изменение скорости носителей заряда в транзисторе при лавинном режиме работы объясняется ограниченной областью объёмного заряда. Исследования работы транзисторов показали, что некоторые сочетания тока и напряжения ведут к смыканию области обеднённого заряда коллекторного и эмиттерного перехода [27]. Таким образом лавинные транзисторы с ограниченной обеднённой зоной имеют малое время нарастания и большие амплитуды формирования импульсов [27]. В случае смыкания области обеднённого заряда разряд лавинного транзистора описывается теорией потокового разряда [40]. Также встречается теория множественного потокового разряда в тиристорах на основе

GaAs [41]. Встречается применение резкого скачка напряжения к неполяризованной слоистой структуре $p^+ – n – n^+$, которая может инициировать распространение ионизационной волны, переводящей структуру в состояние высокой проводимости в течение долей наносекунд. Оказывается, что такое «пробивание» развивается качественно так же, как и в сильно поляризованных структурах в режиме прохождения изолированной волны. Причиной этого сходства является задержка в срабатывании из-за отсутствия начальных носителей в области высокого поля. Во время этой задержки основные носители могут покинуть n-базу и образовать истощённый слой. Аналогичные плоские ионизационные фронты также могут быть возбуждены в плоских объёмных образцах полупроводников без p – n переходов [42].

1.3 Приёмники излучения

Приемник излучения является ключевым компонентом лидара, обеспечивая регистрацию и обработку отраженного лазерного импульса для получения точной информации об окружающей среде и объектах, находящихся на трассе зондирования.

Традиционно, для детектирования слабых световых потоков применяют фотоэлектронные умножители (ФЭУ). Фотоэлектронные умножители используют эмиссию фотоэлектронов из фотокатода в качестве метода детектирования. Квантовая эффективность, то есть вероятность того, что образуется фотоэлектрон при взаимодействии с фотоном, у лучших катодов составляет около 0.4 в диапазоне длин волн от 400 до 500 нм. Объясняется это тем, что фотоэлектроны испускаются во всех направлениях, в том числе и в фотокатод [43]. Благодаря высокому коэффициенту усиления и короткому выходному импульсу, фотоэлектронный умножитель может создавать импульсы тока для отдельных фотонов светового сигнала.

С развитием полупроводниковых технологий, все большие применение стали находить фотодиоды как одиночные, так и матричные. При взаимодействии с

фотоном внутри гетероструктуры полупроводника происходит генерация пары электрон-дырка. Отметим, что квантовая эффективность полупроводниковых детекторов может теоретически быть равной единице. Однако, на практике для длин волн от 520 до 700 нм у современных фотодиодов эффективность составляет около 60% и около 20% для более старых моделей [44]. Эффективность близкая к 90% наблюдается лишь в области длин волн 610 нм [45]. Такая эффективность связана с тем, что ток, вызванный одной парой электрон-дырка, не достаточен для прямого измерения. Поэтому необходим механизм усиления, например лавинный эффект, а он в свою очередь вносит потери в эффективность работы фотодиода. Для работы в лавинном режиме фотодиод подключается к высокому обратному напряжению, таким образом, что носители заряда генерируют новые пары электрон-дырка.

Стандартные лавинные фотодиоды (ЛФД, англ. avalanche photodiode, APD) обеспечивают стабильный коэффициент усиления порядка 10^2–10^3. Однако, коэффициент усиления порядка 10^3 слишком низок для обнаружения одиночных фотонов с требуемым временным разрешением для коррелированного по времени счета одиночных фотонов (от англ. time-correlated single photon counting, TCSPC [46]).

В русскоязычной литературе встречаются похожие термины для обозначения этого метода. В биомедицине для регистрации одиночных фотонов от люминесценции используется термин время-коррелированный счет одиночных фотонов [47, 48, 49]. Также для диагностики барьерного разряда этот метод называется коррелированный по времени счет одиночных фотонов [50].

Более высокий коэффициент усиления обычно приводит к нестабильности, когда лавина становится самоподдерживающейся и разрушает диод. Однако можно избежать разрушения, если использовать схемы активного или пассивного гашения лавины. При этом подходе можно добиться обнаружения одиночных фотонов. Этот принцип часто называют режимом Гейгера [2, 51, 52, 53, 54].

В качестве пассивного гашения лавины в фотодиодах используется последовательное сопротивление и паразитная емкость между резистором и землей. Во время пробоя емкость разрежается и ток разряда проходит через нагрузку и диод, создавая короткий выходной импульс. Напряжение на диоде падает, и лавина гаснет. Восстановление напряжения на диоде до напряжения питания занимает от нескольких сотен наносекунд до нескольких микросекунд. После достижения напряжения пробоя вероятность обнаружения фотона постепенно повышается до исходного уровня.

Для активного гашения используется электронная схема гашения, суть работы которой состоит в следующем [53, 55, 56, 57]. После пробоя, выходной импульс диода запускает схему гашения, которая в течение примерно 20–50 нс временно снижает обратное напряжение диода ниже уровня пробоя. В течение нескольких наносекунд после гашения обратное напряжение быстро восстанавливается. В связи с этим, активное гашение позволяет достичь гораздо более высокой скорости восстановления по сравнению с пассивным гашением. Однако возникают сложности при использовании гасящего импульса к высоковольтной стороне диода без конденсатора или трансформатора. Таким образом, вольтамперная характеристика фотодиода немного флуктуирует выше уровня устойчивого состояния после импульса гашения, а затем стабилизируется. Из-за этого в фотодиодах с активным гашением изменяется эффективность в зависимости от частоты срабатывания. В однофотонных фотодиодах с пассивным гашением средний ток диода ограничивается сопротивлением, что обеспечивает защиту от повреждений. Напротив, в фотодиодах с активным гашением ток не ограничивается автоматически, поэтому может возникнуть перегрузка при многократном срабатывании. Обратное напряжение, подаваемое на однофотонные фотодиоды, может достигать 200–500 В. В результате этого мощность, рассеиваемая в диоде во время лавинного пробоя, становится значительной.

Для уменьшения частоты срабатывания однофотонного фотодиода применятся техника стробирования. В этом режиме обратное напряжение на диод

подаётся импульсом выше уровня пробоя в течение 1–100 нс. Если в этом интервале времени фотон взаимодействует с фотодиодом, развивается лавина. Стробирование позволяет уменьшить шум при зондировании лазерным импульсом за счёт контроля времени работоспособности однофотонного фотодиода. Поэтому эта техника активно применяется для коррелированного по времени счета фотонов. Также лавинные фотодиоды широко используются для экспериментов по квантовому распределению ключей и могут быть использованы для подсчета фотонов с низкой интенсивностью и низкой частотой повторения импульсов [58, 59, 60].

Однофотонные лавинные фотодиоды в независимости от схемы гашения требуют охлаждения для предотвращения генерации лавины в следствии тепловых носителей. Даже охлаждение диода до -30° С не решает проблему, поскольку величина паразитного тока на единицу активной площади на несколько порядков выше, чем у фотоумножителей. Поэтому однофотонные лавинные фотодиоды изготавливаются только с небольшой активной зоной, порядка 10^4 квадратных микрометров.

1.4 Лазерный мониторинг газов и аэрозолей в сейсмически активных областях

Статистический режим работы лидара расширяет возможности измерений в оптически неоднородных средах, что важно для анализа концентрации газов и аэрозольных частиц в атмосфере. Такой подход обеспечивает высокую точность и воспроизводимость результатов, делая его незаменимым инструментом в экологическом мониторинге, метеорологии, и изучении атмосферы. Кроме того, статистический режим лидара позволяет проводить измерения на больших расстояниях и в широком диапазоне метеорологических условий, расширяя границы понимания процессов, происходящих в атмосфере Земли. Данный метод особенно ценен при необходимости изучения динамики распределения газов и аэрозолей на различных высотах, позволяя отслеживать их перемещение и

трансформацию в реальном времени. Этот аспект важен для понимания глобальных климатических изменений, оценки влияния промышленной деятельности на атмосферу и разработки мер по снижению антропогенного воздействия на окружающую среду. Этот метод позволяет выявлять и количественно оценивать распределение различных компонентов воздуха, даже если они распределены неравномерно или находятся в сложных для изучения условиях.

Раннее обнаружение вулканической активности — главная цель вулканологии; однако надёжных методов прогнозирования пока нет [61, 62]. Обычно вулканическая активность сопровождается локальными сейсмическими толчками, а выбросы аэрозоля из трещин могут происходить за несколько часов до извержения [63, 64]. В частности известно что, в результате мониторинга потоков газов (CO_2 и отношения CO_2/SO_2) вулкана Стромболи (Италия) было обнаружено, что динамика эманации этих газов значительно изменяется перед проявлением вулканической активности [65, 66, 67]. Несмотря на многочисленные попытки найти ранние признаки извержений и землетрясений, до сих пор отсутствуют надёжные методы прогнозирования подобных явлений [61, 62]. При краткосрочном прогнозировании извержений было обнаружено, что непрерывный мониторинг выхода вулканических газов (H_2O, SO_2 и CO_2) может служить индикатором надвигающегося извержения [68, 69, 70] появляющимся за несколько часов до извержения, в то время как сейсмометры не дают никаких признаков тектонической активности. Кэндзю и др. [71] изучали потоки углекислого газа на вулкане Фудзи и было обнаружено, что увеличение выброса диоксида углерода обусловлено предвулканичной дегазацией коры Земли. Однако, измерение выбросов магматических газов может быть сильно затруднено, так как точки выхода газов могут изменять свое положение вследствие сейсмических событий, что требует построения сети сенсоров [67]. Хорошим подходом для решения данной проблемы является использование методов дистанционного зондирования. Спутниковое зондирование не может обеспечить необходимой чувствительности

детектирования тектонических газов, поэтому лазерное дистанционное зондирование является лучшим решением. Например, Фиорани и др. [63, 72] использовали лидар для мониторинга дегазации углекислого газа в фумаролах вулканов Флегрейских полей и вулкана Этна. Они показали, что лидар дифференциального поглощения (DIAL) может удаленно количественно измерить выброс углекислого газа с вулканов с высоким временным разрешением в несколько минут и ошибкой менее 30% [73].

Аэрозольный лидар, работающий в режиме счета фотонов, представляет собой перспективный инструмент для мониторинга атмосферы, способный принести значительную пользу при прогнозировании вулканической активности. Квайсер и др. [74] разработали лидар на основе волоконного лазера для дистанционного зондирования углекислого газа, но для зондирования необходимо было устанавливать ретроотражатель вблизи вулкана, что исключало долговременные измерения (из-за загрязнения поверхности ретроотражателя).

Ещё одно применение мониторинга деятельности вулканов предложено Сассеном и др. [64], которые использовали поляризационный лидар для диагностики вулканических пепельных аэрозолей различных типов, которые представляют угрозу для воздушного трафика самолетов.

Осуществляя замеры аэрозольного состава воздуха, лидар позволяет отслеживать изменения, вызванные выбросом газов из недр Земли, которые часто являются предвестниками вулканических извержений. Это делает его неоценимым инструментом для раннего обнаружения признаков вулканической активности, тем самым улучшая возможности прогнозирования извержений и позволяя предпринимать меры для минимизации потенциального ущерба и рисков для населения. Важно отметить, что применение коротких лазерных импульсов в такой системе увеличивает точность локализации источников эманации газов, обеспечивая дополнительные данные для анализа. Использование лидара в режиме счета фотонов для изучения атмосферы вблизи вулканов предоставляет

уникальную возможность для детального анализа изменений в составе атмосферы, которые могут указывать на увеличение вулканической активности.

Глава 2. Принципы зондирования диодным лазером с пикосекундным фронтом в статистическом режиме

2.1 Инжекционный диодный лазер с пикосекундной ступенькой на фронте наносекундного импульса

Одним из направлений представленной работы является разработка способа сокращения длительности фронта наносекундного оптического импульса диодного лазера, что можно применить для повышения точности лазерной дальнометрии. В связи с этим было проведено исследование генерации лазерного инжекционного диода при накачке токами с пикосекундными фронтами. Для управления формой питания лазерного диода был разработан генератор на базе биполярного транзистора FMMT417 с рабочим напряжением до 300 В. После анализа доступных промышленных компонентов был выбран лазерный диод SPL PL90_3 (OSRAM), собранный из трёх AlGaAs-диодов по стековой технологии в пластиковом корпусе без встроенного генератора тока. Рекомендованный режим работы этого диода предполагает накачку током и генерацию лазерного импульса 100 нс. Выбор данного диода для лидара в режиме счета фотонов обусловлен следующим:

1. Длина волны генерации должна быть в ближнем ИК диапазоне;
2. Диод должен быть собран по стековой технологии для достаточной энергии наносекундных импульсов;
3. Малый размер излучающей области лазера 200x10 мкм должен позволять генерировать импульсы с пикосекундными фронтами.

Микроконтроллер ATMEGA8515 (DD1) принимает стартовый импульс (START), затем формирует импульс тока до 24 мА с крутым фронтом. Далее на эмиттерном повторителе VT1 этот импульс усиливается до 250 мА. И подаётся на базу биполярного транзистора FMMT417 (VT2) в лавинном режиме работы. Питающий конденсатор C4 через биполярный транзистор разряжается на инжекционный диодный лазер (VD2). Формируется оптический импульс, повторяющий ток разряда.

После выключения транзистора VT2 конденсатор C4 заряжается через резисторы R5 и R6. После заряда конденсатора C4 генератор импульсов тока готов к формированию следующего импульса лазера. Для работы микроконтроллера используется линия питания 5В. А для накачки конденсатора используется высоковольтное напряжение +HV, значение которого выбирается в процессе настройки генератора.

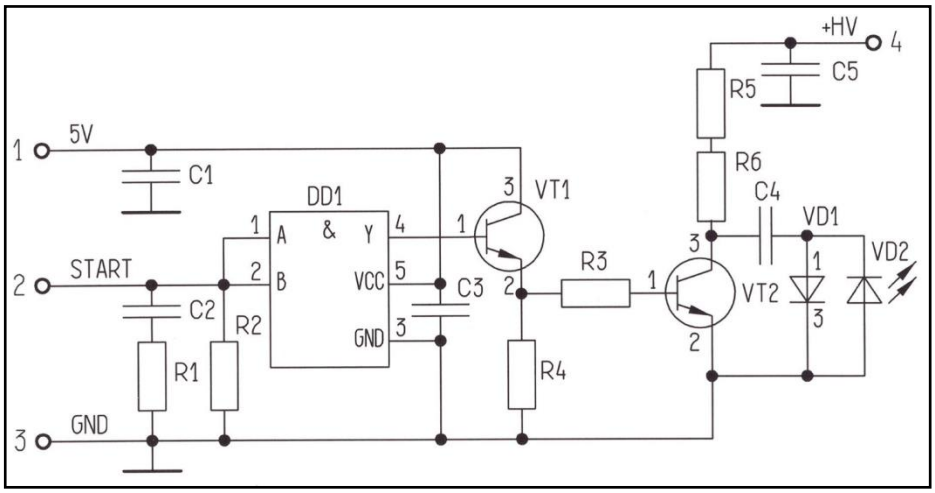

Рисунок 2. Электрическая схема генератора импульсов тока для инжекционного диодного лазера VD2. DD1 – микроконтроллер ATMEGA8515, VT1 – эмиттерный повторитель для усиления, VT2 – биполярный транзистор FMMT417 в лавинном режиме для коммутации разрядного тока конденсатора C4 и инжекционного диодного лазера, +HV – напряжение питания до 300 В, R5 и R6 – согласованная нагрузка для заряда конденсатора C4, VD1 – диод, обеспечивающий зарядку конденсатора C4.

Излучаемую энергию лазера можно рассчитать по величине энергии, запасённой в конденсаторе C4. Используя формулу $W = CU^2/2$, получаем W = 33,7 мкДж. Так как КПД импульсного лазера составляет около 5%, получаем энергию в импульсе около 1,7 мкДж.

Проведены измерения параметров лазерного импульса на цифровом осциллографе (7704B, Sony Tektronix, США) с полосой пропускания 7 ГГц и фотодиоде (5 ГГц, Thorlabs DET08C(/M)). На Рисунок 3 показана зависимость

сокращения длительности наносекундного лазерного импульса при уменьшении ёмкости. Отметим, что даже при малой накачке (ёмкость 82 пФ) удалось добиться устойчивой генерации импульсов длительностью 2.7 нс.

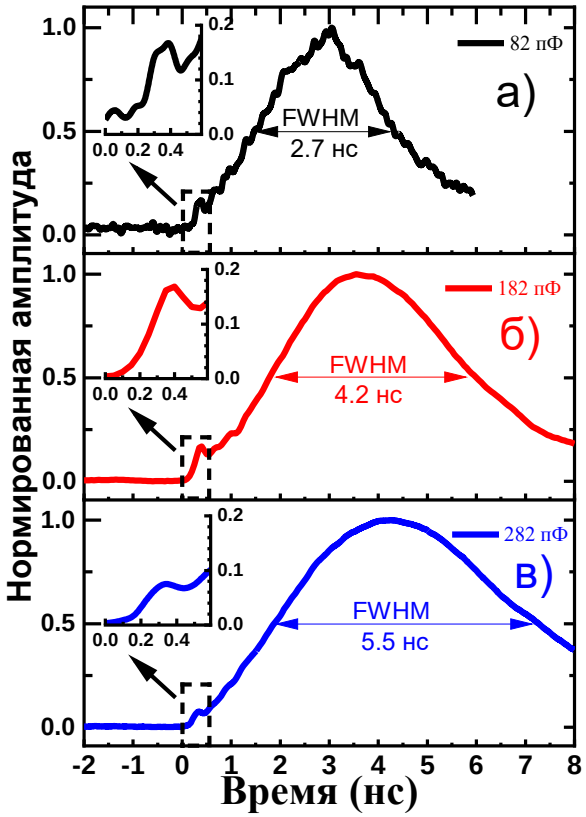

Рисунок 3 Осциллограммы импульсов генерации диодного лазера при увеличении значения разрядной ёмкости: (а)- 82, (б) - 182 и (в) - 282 пФ при напряжении заряда 275 В. FWHM (англ. full width at half maximum) - полная ширина на уровне половины высоты

Из осциллограммы импульса (Рисунок 3б, вставка) видно, что длительность ступеньки на фронте импульса составляет около 200 пс. Если учесть полосу фотоприёмника, которая не превышает 5 ГГц, можно ожидать, что длительность фронта ступеньки может быть и менее 200 пс. Заметим, что в литературе ранее уже наблюдали подобную ступеньку на фронте наносекундного лазерного импульса [75], однако авторы никак не комментировали появление подобной форму фронта лазерного импульса. Совокупность этих данных даёт основание утверждать, что обнаружен новый режим генерации наносекундных импульсов специфической

формы (со ступенькой вначале) диодными лазерами (как коммерческими, так и с плотным монтажом элементов) с пикосекундным (≤200 пс) фронтом. Далее отметим, что, поскольку лазеры являются инжекционными, то излучение отражает форму импульса тока разрядного контура с быстрыми ключами на биполярных лавинных транзисторах. Действительно, недавно [27] было установлено, что современные быстрые ключи на этих элементах обеспечивают генерацию импульса тока с пикосекундной ступенькой в начале[76].

2.2 Адаптационный режим лазерного зондирования в режиме счета фотонов

Факт наличия ступеньки на фронте лазерного импульса порядка сотен пикосекунд может играть нам на пользу при зондировании. Гистограмма обратного рассеяния фотонов по времени при зондировании плоской однородной мишени лидаром в режиме счета фотонов фактически представляет собой оцифровку лазерного импульса. Согласно методу счета фотонов необходимо соблюдение условия вероятности регистрации обратно рассеянных фотонов много меньше единицы (режим «слабого сигнала»). Если количество зарегистрированных фотонов будет приближаться к количеству лазерных импульсов (вероятность регистрации импульса близка к единице – приближение «сильного» сигнала), то их распределение на гистограмме перестанет отражать форму зондирующего импульса. При этом вероятность регистрации фотонов от переднего фронта лазерного импульса будет выше, чем от спада [77]. При увеличении вероятности регистрации обратно рассеянных фотонов (увеличение площади приемной антенны, увеличение альбедо удаленного объекта и т.д.) точность дальнометрии режиме «сильного» сигнала будет возрастать вплоть до разрешающей способности временно-цифрового преобразователя, что пропорционально размеру ячейки на гистограмме. При этом важно отметить время развития лавины на однофотонном лавинном фотодиоде, оно должно быть меньше разрешающей способности на временно-цифрового преобразователя. Отметим, что для однофотонного

лавинного фотодиода время развития лавины обычно меньше 1 нс [78]. Для проверки этого факта был проведён эксперимент с плоской мишенью при разной «степени загрузки» приёмника [79]. Плоскую мишень установили на расстоянии 35 метров от созданного лидара. Для формирования гистограммы лидар собирал данные по 10^4 импульсам лазера. Вероятность регистрации обратно рассеянных фотонов регулировали с помощь ирисовой диафрагмы на приёмном объективе, которая ограничивала поле зрения однофотонного лавинного фотодиода. Вероятность срабатывания фотоприёмника измерялась по отношению числа принятых фотонов к числу стартовых импульсов лазера. На Рисунок 4 представлена временная форма лазерного импульса, а также гистограммы фотонов обратного рассеяния в режимах «слабого» и «сильного» сигналов.

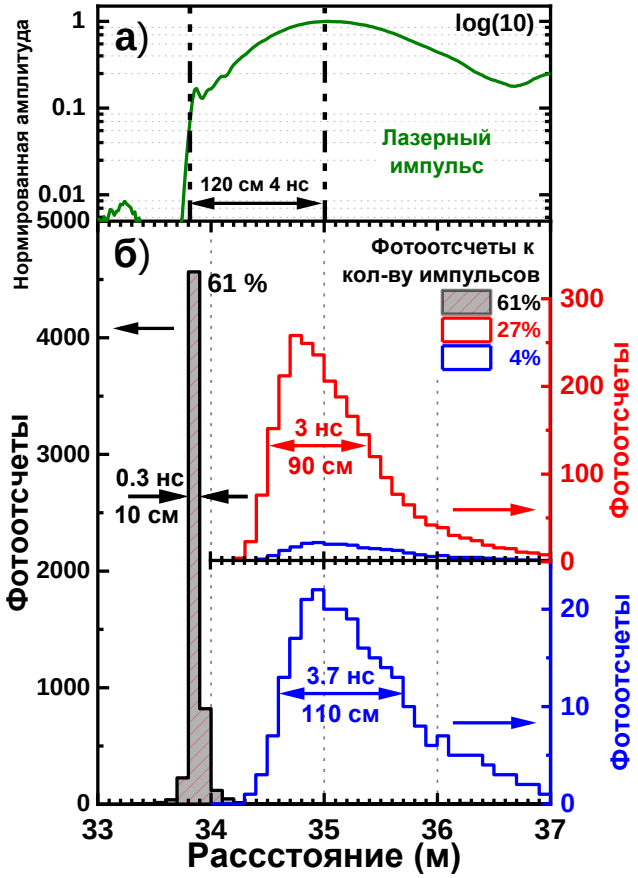

Рисунок 4 Результаты лидарного зондирования в адаптационном режиме для плоской мишени на расстоянии 35 метров. (а) - осциллограмма импульса лазера в логарифмическом масштабе по оси ординат и с пунктирными линиями на

расстоянии от фронта-ступеньки до максимума импульса. (б) – изменение гистограммы распределения фотоотсчётов по интервалам времени (расстояния) в зависимости от вероятности регистрации фотонов

Из Рисунок 4 видно, что увеличение количества зарегистрированных обратно рассеянных фотонов сопровождается изменением их распределения по интервалам времени (расстояния). Количество зарегистрированных фотонов от передней границы лазерного импульса возрастает вплоть до 20-кратного увеличения, а лидар переходит в режим работы «сильного» сигнала. В тоже время функция распределения зарегистрированных фотонов сужается до нескольких ячеек на гистограмме, а также смещается более чем на 1 м к фронту ступеньки вначале импульса (см. Рисунок 4б). Размер (120 см) между двумя вертикальными пунктирными линиями показывает расстояние между максимумом 4 нс импульса (осциллограммы) и началом 200 пс фронта импульса лазера.

Отсюда следует, что на коротких расстояниях (метры), когда вероятность регистрации обратно рассеянных фотонов значительно возрастёт, лидар переходит в режим работы «сильного» сигнала, что приведёт к уменьшению распределения интервалов времени задержки между стартом генерации лазерного импульса и моментом срабатыванием фотодиода, вплоть до одной ячейки на гистограмме. Фактически, в таком режиме работы лидар переходит в режим работы дальномера с повышением точности определения расстояния равной разрешению временно-цифрового преобразователя. В этом случае наличие ступеньки на фронте лазерного импульса позволяет повысить разрешающую способность по дальности измерения, поскольку более резкая граница увеличивает вероятность регистрации обратно рассеянных фотонов именно от неё. Например, в текущей работе показано, что для 3 нс лазерного импульса, наличие ступеньки в 200 пс позволяет в 10 раз увеличить точность определения расстояния лидаром.

Адаптационное управление работой лидара являются критическими важным при стыковке платформ в космическом пространстве. Здесь поиск причала на большом удалении проводят в режиме «слабого сигнала» с длинным (100-500 м)

стробом и, напротив, сближение и стыковку проводят в режиме «сильного сигнала» с сантиметровом разрешением расстояния и скорости сближения около 1 см/с. Особым преимуществом здесь является цифровой способ обработки рассеянных фотонов, который позволяет вести зондирование без «мёртвой зоны» вплоть до касания. Отметим, что однофотонный лавинный фотодиод устойчив к повреждению приёмной аппаратуры мощным обратным сигналом или бликом от поверхности объекта зондирования. Последний фактор допускает использование светоотражающих покрытий поверхности причала, которые кратно повышают значение отношения сигнал/шум и, соответственно, дальность обнаружения и зондирования. Несомненно, что предложенный адаптационный подход работы лидара является важной альтернативой использования дорогостоящих пикосекундных лазеров в лидарах, а, значит, делает диодные лидары более доступными для различных научных и технических приложений.

Глава 3. Экспериментальная часть

3.1 Лидар с неодимовым лазером на второй гармонике и пассивной модуляцией добротности для подводного зондирования

Лазерное дистанционное зондирование водных объектов является значительно более сложной задачей по сравнению с атмосферными измерениями, что связано с большей оптической плотностью среды, большими флуктуациями показателя преломления вследствие градиентов физических параметров в натурном эксперименте (градиенты температуры, солености, концентрации частиц и пузырьков и т.д.), высокими коэффициентами поглощения и рассеяния в воде. Для задач дистанционного зондирования водной среды важно чтобы коэффициент поглощения лазерного излучения был минимальным с тем, чтобы максимально увеличить трассу зондирования. Согласно измерениям коэффициента поглощения в диапазоне длин волн от 380 до 700 нм [80], окно прозрачности в видимом диапазоне расположено от 430 до 550 нм, поэтому излучение второй гармоники неодимового лазера является подходящим так как для длины волны 532.5 нм коэффициент поглощения составляет 0.0447 ± 0.0017 м$^{-1}$ [80]. К другим преимуществам, важным для разработки подводного лидара, относятся высокая стабильность и надёжность работы лазера при эксплуатации в полевых условиях, а также его компактные размеры. Как правило подобные лазеры имеют высокое качество пучка $M^2 \sim 1$ [81], что важно для повышения пространственного разрешения в режиме сканирования. Длительность лазерного импульса составляет от сотен пикосекунд до десятков наносекунд, что будет определять точность дальнометрии [82]. Отметим, что при использовании пассивного модулятора добротности отсутствуют высоковольтные наводки, что упрощает электрические схемы питания, так как нет необходимости формировать несколько линий питания для разных компонентов лидара. К недостаткам можно отнести нестабильность момента генерации лазерного импульса при пассивной модуляции добротности, в

следствие чего необходима сложная схема синхронизации фотоприемника для проведения точной дальнометрии.

Для подводного зондирования был создан макет лидара [83, 84], где в качестве передатчика выбрали Nd:АИГ лазер со второй гармоникой лазера и с пассивной модуляцией добротности с продольной накачкой диодным лазером. Излучение Nd:АИГ лазера преобразовывали в нелинейно-оптическом кристалле (KTP) во вторую гармонику (532 нм). На выходе удвоителя блокировали (светофильтром) излучение лазера на основной длине волны 1064 нм и пропускали излучение второй гармоники.

Лазер генерировал импульсы длительностью 1.5 нс с частотой повторения 4-5 кГц и энергией импульса до 20 мкДж при максимальной накачке лазера (Рисунок 5).

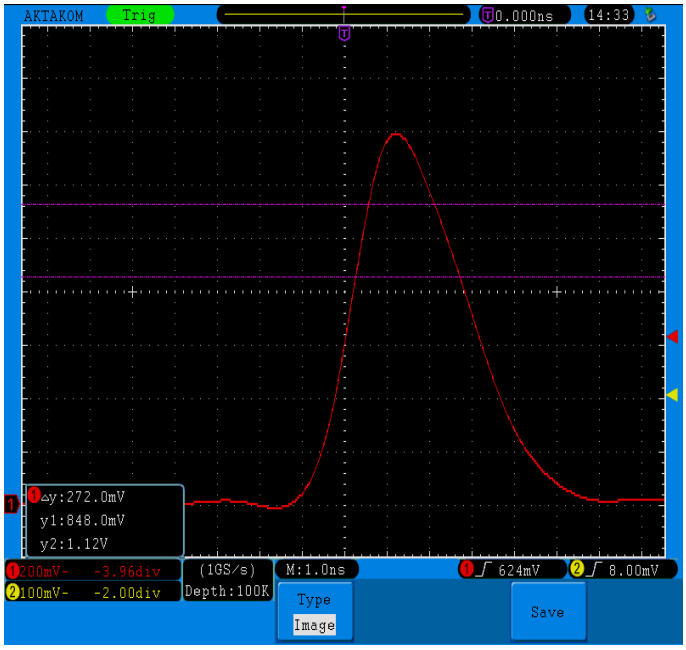

Рисунок 5 Форма лазерного импульса, измеренная на осциллографе ADS-2332 Актаком (полоса пропускания 300 МГц) с помощью фотодиода 11 HSP-FS Standa (1 ГГц)

Для продления ресурса работы лазера мы уменьшали накачку так, чтобы энергия импульсов второй гармоники не превышала 1-2 мкДж при полностью

открытой апертуре (диаметр 3 см) выходного объектива. Для работы на коротких дистанциях до 200 метров мы уменьшали энергию излучения лазера, регулируя диаметр апертуры с помощью ирисовой диафрагмы в пределах 1-8 мм.

В качестве приёмника излучения был использован кремниевый однофотонный лавинный фотодиод с диаметром приёмной площадки 40 мкм, в который интегрирован охладитель на основе элемента Пельтье. На приёмник подавали последовательность стробов-импульсов питания приёмника длительностью 30 нс. Апертура антенны (диаметр объектива) приёмника составляла 12 мм, но также варьировалась в пределах 0,5-4 мм ограничивающей ирисовой диафрагмой при зондировании на коротких трассах. Фотография лидара в сборе представлена на Рисунок 6.

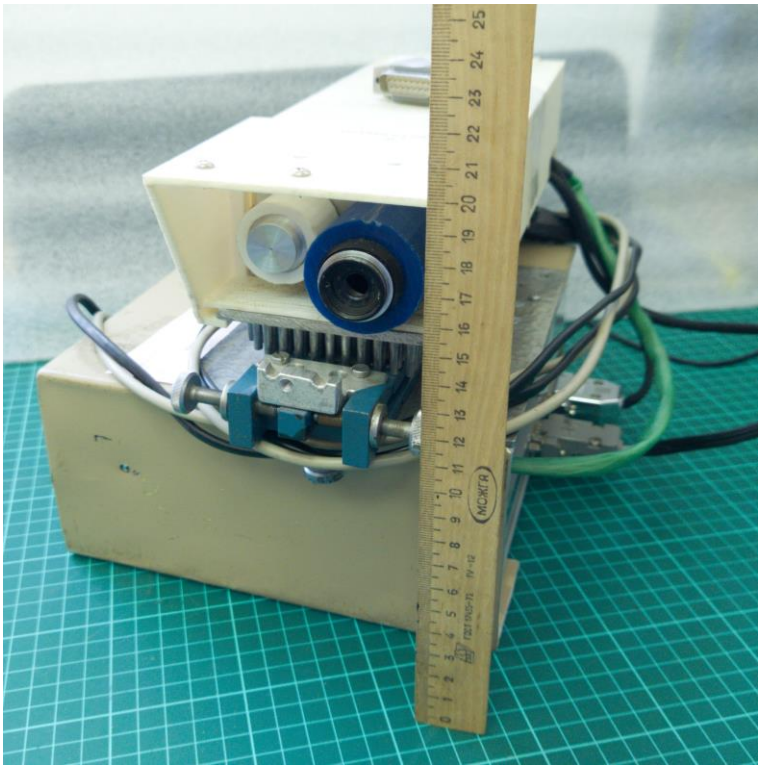

Рисунок 6 Лидар с неодимовым лазером на второй гармонике и пассивной модуляцией добротности для подводного зондирования

Общий принцип работы разработанного лидара соответствует описанному в разделе 1.1 статистическому способу работы, но имеет ряд особенностей. Измерение временной задержки производится микросхемой временно-цифрового

преобразователя (модель TDC-GP1, ACAM). Разрешающая способность которого составляет 250 пс. Микросхема TDC-GP1 использует частоту 5 МГц кварцевого генератора для калибровки времени задержки методом подстройки напряжения питания измерительного узла с помощью фазовой автоподстройки частоты.

Принцип работы лидара на лазере с пассивным модулятором добротности

Для управления режимом работы однофотонного лавинного фотодиода используют сигнал CLOCK (см. Рисунок 7 и Рисунок 8) частотой 2,5 МГц. При низком уровне сигнала CLOCK на однофотонный лавинный фотодиод подаётся пороговое напряжение, при высоком уровне CLOCK на приёмнике нет порогового напряжения. Измерение временной задержки срабатывания однофотонного лавинного фотодиода начинается после прихода стартового сигнала (START), который формируется PIN фотодиодом от блика лазерного импульса. Возможно два варианта событий:

1) сигнал START может прийти в момент, соответствующий нижнему уровню сигнала CLOCK, тогда на фотодиод подано пороговое напряжение и фотон с долей вероятности может быть зарегистрирован (см. Рисунок 7);

2) сигнал START приходит в момент «высокого уровня» сигнала CLOCK, тогда на фотодиоде отсутствует пороговое напряжения и фотон не может вызвать развитие лавины (см. Рисунок 8).

Микроконтроллер ограничивает количество пороговых импульсов на однофотонном лавинном фотодиоде 4 временными интервалами (GATE) «низкого уровня», при которых возможно зарегистрировать фотон однофотонным приёмником, который формирует стоп импульс (STOP) для остановки измерениям времени счётчиком. Таким образом, максимальный диапазон измерения составляет 240 м в воздухе, если сигнал START придет сразу после перехода сигнала CLOCK к нижнему уровню, или 180 м в воздухе, если сигнал START придет непосредственно перед положительным фронтом сигнала CLOCK. Поскольку частоты сигналов START и CLOCK не синхронизированы между собой, то не

получится синхронизировать сигнал GATE, а значит вероятность срабатываний фотоприёмника будет одной и той же для всей длины трассы измерений.

Время восстановления однофотонного фотодиода после регистрации фотона («мертвое время») составляет 200 нс. Если во время этого интервала происходит формирование стартового импульса, то фотоприёмник не регистрируем фотоны до начала следующего спада напряжения CLOCK. Таким образом, «мертвое время» лавинного фотоприёмника варьируется от 200 нс (если срабатывание произошло в конце интервала по дальности 30 м) до почти 400 нс (если срабатывание произошло в начале интервала по дальности 30 м).

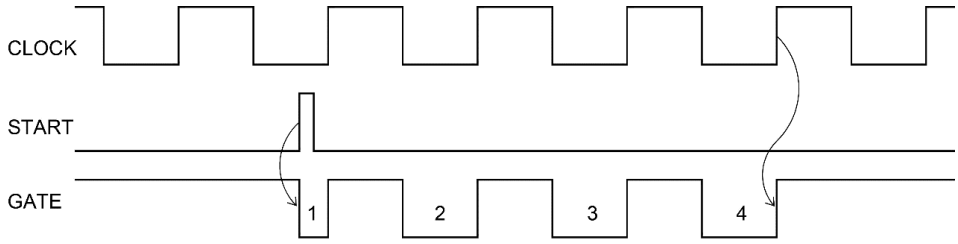

Рисунок 7 Временная диаграмма возможности регистрации обратно рассеянного фотона относительно стартового импульса (START), когда стартовый импульс приходит во время подачи напряжения на однофотонный лавинный фотодиод. Строб (GATE) – временной интервал, соответствующий измерению расстояния 180 м в воздухе. CLOCK – меандр подачи порогового напряжения на однофотонный лавинный фотодиод с частотой 5 ГГц.

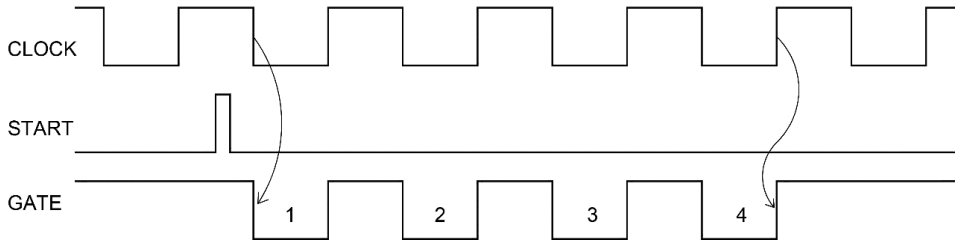

Рисунок 8 Временная диаграмма возможности регистрации обратно рассеянного фотона относительно стартового импульса (START), когда стартовый импульс приходит до подачи напряжения на однофотонный лавинный фотодиод. GATE – временной интервал соответствующий измерению расстояния 180 м в воздухе. CLOCK – меандр подачи порогового напряжения на однофотонный лавинный фотодиод с частотой 5 ГГц.

Поскольку лазер работал в режиме пассивной модуляции в качестве стартового импульса START использовался фотоприемник установленный непосредственно в объективе после кристалла KTP.

3.2 Аэрозольный лидар на инжекционном диодном лазере

Для зондирования туманов и аэрозолей был разработан лидар с использованием описанных во второй главе схем и компонентов.

В качестве диодного лазера использовался SPL PL90_3 (OSRAM), собранный из трёх AlGaAs-диодов по стековой технологии в пластиковом корпусе без встроенного генератора тока. Для питания лазера в генераторе тока использовался конденсатора с емкостью 85 пФ. Соответственно, диодный лазер генерировал оптические импульсы длительностью 2.7 нс с пикосекундной ступенькой на фронте. Детальное описание работы лазерного диода приведено в разделе 2.1.

В качестве диодного лазера использовался SPL PL90_3 (OSRAM), собранный из трёх AlGaAs-диодов по стековой технологии в пластиковом корпусе без встроенного генератора тока. Для питания лазера в генераторе тока использовали конденсатор с ёмкостью 85 пФ. Соответственно, диодный лазер генерировал оптические импульсы длительностью 2.7 нс с пикосекундной ступенькой на фронте. Детальное описание работы лазерного диода приведено в разделе 2.1.

В качестве приёмника рассеянного излучения был выбран фотодиод SAP-500-T6 (Laser Components). Используемый однофотонный кремниевый лавинный фотодиод (SPAD), обладает необходимыми характеристиками для разрабатываемого лидара [54]:

1- время развития лавины не превышает 500 пс;

2- Гейгеровским режимом счёта фотонов;

3- отсутствием повреждений в условиях большой оптической нагрузки при работе в режиме дальномера;

4- короткое (не более 100 нс) время «восстановления» приёмника.

5 – высокой квантовой эффективностью в видимом диапазоне длин волн.

В качестве измерителя времени задержки между стартом оптического импульса и моментом срабатывания однофотонного лавинного фотодиода был выбран временно-цифровой преобразователь (TDC-GP1, ACAM). Микросхема TDC-GP1 требует калибровки с помощью фазовой автоподстройки частоты, что позволяет повысить точность измерений, а также снизить влияние температуры на стабильность измерений. Для этих целей используется кварцевый генератор с частотой 5 МГц.

Для управления и измерений была разработана интегральная плата на основе микроконтроллера ATMEGA8515. Краткое описание схемы работы описано далее. Измерение временных задержек срабатываний фотоприёмника начинается после импульса START, формируемого микроконтроллером в соответствии с программой работы. Одновременно или с опережением на 67 нс в зависимости от выбранного режима работы формируются импульсы LSTART и LCCH. Первый импульс необходим для запуска лазера, второй импульс требуется для выключения тока заряда накопительного конденсатора на время восстановления разрядного ключа генератора тока питания лазера. Микроконтроллер также формирует последовательность временных интервалов импульсами GATE «низкого уровня», во время которых при поступлении импульсов STOP измеряются их временные задержки относительно импульсов START. Число импульсов GATE может быть выбрано программное от двух до четырех в зависимости от необходимого режима работы лидара. Таким образом, максимальное число зарегистрированных событий (по одному в интервале GATE) может равняться 4. Последний интервал GATE формируется со значительной задержкой относительно лазерного импульса и служит для измерения уровня шума, обусловленного темновым шумом и внешней засветкой. После завершения последнего интервала GATE микроконтроллер считывает значения полученных задержек и передает их на компьютер через последовательный порт.

Параметры лидара сведены в таблице (Таблица 2).

Таблица 2. Параметры лидара в режиме счета фотонов на основе диодного лазера в импульсном режиме работы версия 1

Устройство	Описание
1. Передатчик	
Тип лазера	Полупроводниковый, AlGaAs
Энергия импульса	0.3-1 мкДж
Длина волны	910 нм

Импульсная мощность	до 1000 Вт
Длительность импульса	2.7 нс
Апертура передатчика	15х40 мм
Расходимость пучка	3х10 мрад
Частота повторения импульсов	8 кГц
2. Приемное устройство	⌀12 мм
Входная апертура объектива	Кремниевый однофотонный лавинный диод (SPAD)
Тип детектора	10 нм
Ширина спектральной полосы пропускания фильтра	

Масса лидара не превышает 400 грамм: 100 г. – приёмо-передающий оптический блок и 300 г. – интерфейсный блок.

Для проверки работоспособности лидара был получен ряд гистограмм с 10 параллельными измерениями на каждую точку и показана зависимость измеренного расстояния от дальности смещения мишени (Рисунок 9 б). Измеренное по сравнению с заданным смещение было аппроксимировано линейной функцией, после чего были построены верхние и нижние доверительные интервалы для определения погрешности измерения расстояния. В лазерной физике принято использовать доверительный уровень вероятности равным 0.95 («критерий 2σ»), поэтому для расчёта доверительных интервалов выбрали это значение. Точность лазерного измерения расстояния (точность дальнометрии) определили как соответствующий интервал абсцисс между верхними и нижними кривыми доверительных интервалов для выбранного значения ординаты (см. Рисунок 9 б) [85]. Согласно такому определению, точность дальнометрии зависит от величины расстояния до мишени, однако для разработанного прибора эта зависимость была слабой (от ±0,17 до ±0,21 м в зависимости от удаления мишени).

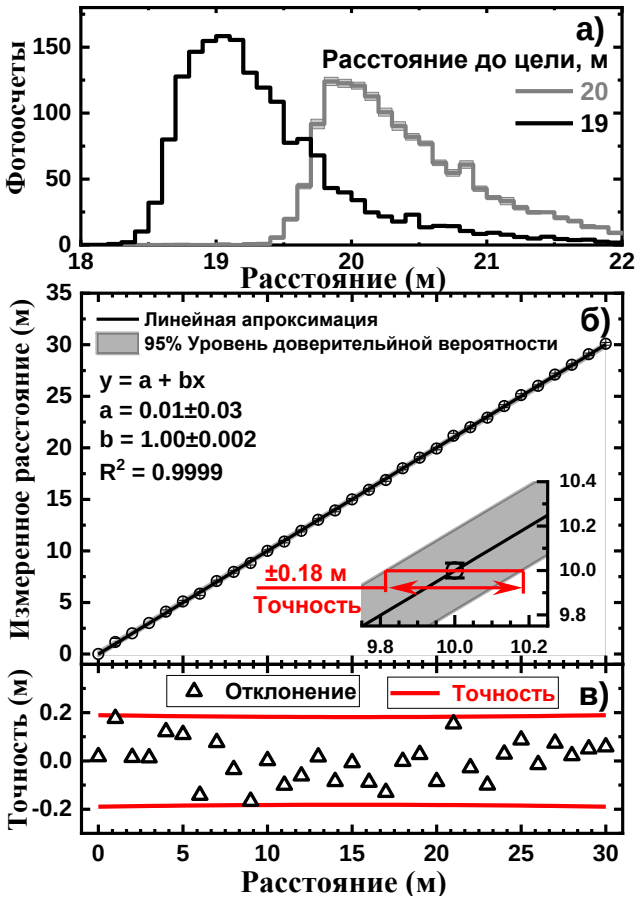

Рисунок 9 (а) - Гистограммы обратного рассеяния для целей, смещенных на расстояние 19 (черный) и 20 (серый) м от лидара; (б) - измерение расстояния до целей на основе гистограмм обратного рассеяния; (в) - точность лазерного дальномера (красные сплошные линии) и отклонения (черные пустые треугольники).

Трехмерная модель-сборка и фотографии разработанного в ИОФ РАН аэрозольного лидара представлены на (Рисунок 10). Параметры лидара представлены в таблице (Таблица 3).

Рисунок 10 Разработанный в ИОФ РАН аэрозольный лидар в статистическом режиме работы. (а) трёхмерная модель оптического блока лидара; (б) фотография лидара в сборе без корпуса

Таблица 3. Параметры диодного лазера и однофотонного приёмника, используемых для аэрозольного лидара в режиме счета фотонов

Излучатель	
Тип лазера	Полупроводниковый, AlGaAs
Энергия импульса	0.3 мкДж
Длина волны	910 нм
Импульсная мощность	до 200 Вт
Длительность импульса	2.7 нс
Апертура передатчика	22x50 мм
Расходимость пучка	2x7 мрад
Частота повторения импульсов	8 кГц
Приёмник	
Входная апертура объектива	⌀25 мм
Тип детектора	Кремниевый однофотонный лавинный диод (SPAD)
Диаметр чувствительной	500 мкм закрыт диафрагмой

области	517x75 мкм
Ширина спектральной полосы пропускания фильтра	10 нм

Для ограничения поля зрения приемник была рассчитана и изготовлена прямоугольная диафрагма размером 515x75 мкм. Диафрагму изготовили методом лазерной резки фемтосекундными импульсами. Диафрагма была установлена перед однофотонным приёмником. После нескольких сеансов отработки режимов для изготовления пробных диафрагм были созданы несколько диафрагм заданной формы (Рисунок 11). Размеры и качество каждой из диафрагм были проконтролированы на оптическом микроскопе с 8x объективом.

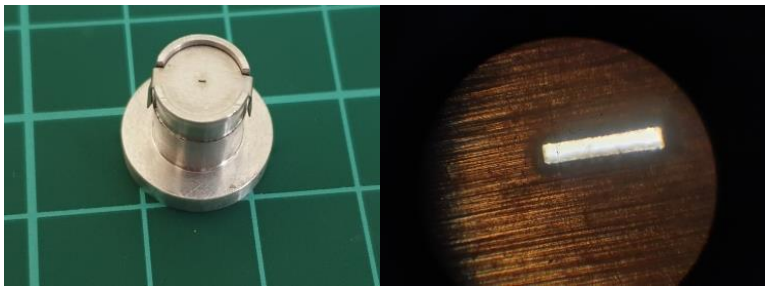

Рисунок 11 Фотография в держателе (слева) и детальное изображение на микроскопе с 8x объективом (справа). Размер диафрагмы 515x75 мкм.

Диафрагма обеспечивает формирование поля зрения приёмника прямоугольного сечения с длинной стороной вдоль горизонтали. Такая диафрагма эффективно уменьшает шумовую засветку приёмника, что приводит к повышению отношения сигнал/шум, а также защищает приёмник от фотонов боковых лепестков диаграммы направленности пучка лазера–передатчика в обратно рассеянном излучении [86]. Выбранная ориентация диафрагмы обусловлена геометрией компоновки лидара, когда каналы лидара расположены горизонтально. В такой схеме лидара и формы диафрагмы функция перекрывания пучка лазера и поля зрения приёмника обеспечивает регистрацию фотонов, рассеянных аэрозолями с меньшего расстояния, что повышает чувствительность лидара.

Для взаимодействия с микроконтроллером лидара по COM порту по стандарту RS-422, была разработана программа на языке программирования Python (Приложение 1). Программа состоит из нескольких модулей, каждый из которых отвечает за свою функцию. Основной модуль main.py отвечает за отрисовку графического интерфейса программы, за выбор протокола взаимодействия с лидаром, ввод параметров измерения и получение массива данных задержек между стартом генерации лазерного импульса и формированием лавины на однофотонном фотодиоде. Модуль main_cli.py использует похожий функционал, но использует для взаимодействия с пользователем интерфейс командной строки. Модуль com_port.py отвечает за взаимодействие с лидаром по COM порту. Модуль measure.py отвечает за передачу параметров измерения в микроконтроллер лидара и формирует массив времен задержки, записывает полученные данные. Модуль project_log.py отвечает за журналирование процесса работы программы. В модуле protocols.py описаны команды для управления микроконтроллером лидара. Модуль read_file.py отвечает за расшифровку данных, записанных в файл в ходе работы модуля measure.py.

Глава 4. Зондирование многослойных сред лидаром в статистическом режиме работы

4.1 Лазерное зондирование через водную среду

Лазерное дистанционное зондирование водных сред представляет огромный интерес для различных областей (от климатологии до оперативного обнаружения подводных аппаратов), однако при измерениях на протяженных подводных трассах (сотни метров и более) сложно добиться высокой достоверности измерений, вследствие малой величины соотношения сигнал-шум, что вызвано различными физическими процессами при распространении лазерного излучения в воде (рассеяние на пузырьках и твердых частицах, вариация оптической плотности вследствие градиентов температуры и солености воды и др.). Важным фактором, ограничивающим широкое применение лидаров является требование безопасности для глаз, которое является еще более строгим при зондировании водных объектов, вследствие высокой эффективности рассеяния вблизи лидара.

Представляется целесообразным оценить возможности разработанного в ИОФ РАН лидара для зондирования многослойных подводных сред. Разработанный лидар (см. раздел 3.2) с неодимовым лазером на второй гармонике и пассивной модуляцией добротности для подводного зондирования должен был обеспечить сантиметровое пространственное разрешение по трассе длиной порядка 200 м [87, 88, 89]. Требовалась проверка это в модельном эксперименте, так как вода сложным объектом для оптического зондировании в следствие флуктуации физико-химических свойств (температура, соленость, наличие пузырьков, твердых частиц и др.) [90, 91, 92, 93].

Модельный эксперимент проводили в лаборатории в пластиковой трубе диаметром 110 мм и длиной 9 м, заполненной водой из городского водопровода (Москва). Труба была составлена из секций с открытыми колодцами в соединениях секций. Входной и выходной торцы трубы оборудованы оптическими окнами, которые обеспечивали два сквозных прохода импульса через 9 м слой: до стены за

трубой и обратно в приёмник лидара. Конструкция трассы позволяла вносить в лазерный пучок полупрозрачные предметы (пластиковые сетки с размером ячеек 3х3мм), которые имитировали природные или искусственные объекты дополнительных потерь к рассеянию в объёмной воде по трассе зондирования (Рисунок 12).

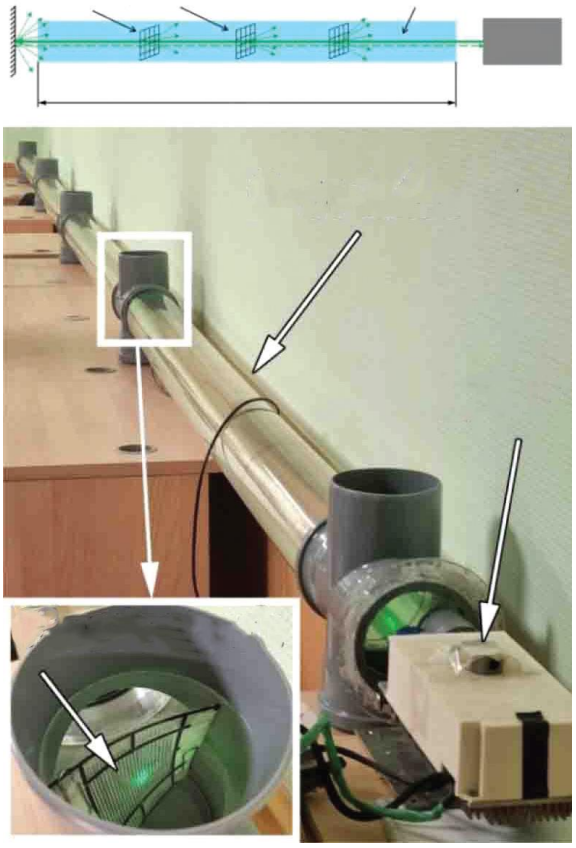

Рисунок 12 Схема подводной трассы длиной 9 м с тремя рассеивающими сетками-препятствиями и общий вид трубы-трассы с примером фото рассеивающей сетки,

На Рисунок 13 изображен характерный вид гистограммы эхо-сигнала при распространении импульсов лидара вдоль трассы, полученный в ходе лабораторных экспериментов. Из рисунка видно, что рассеивающие препятствия регистрируются лидаром на расстояниях их последовательного размещения от входного окна трубы, наполненной водой. Особый интерес вызывает большая амплитуда сигнала рассеяния на стене за трубой после прохода импульса лидара

через слой воды толщиной 9 м. Отсюда следует, что, несмотря на предельно малую энергию импульсов (1-2 мкДж), лидар обеспечил мониторинг рассеивающих объектов вдоль трассы, а также непрозрачных препятствий (стена на удалении ~10 м) на двойном проходе через слой воды 9 м. При этом было достигнуто отношение сигнал/шум ~35 (140 отсчётов от стены на 10 м и 4 до неё, на 9.5 м, см. Рисунок 13).

Оценим предельную дальность обнаружения удаленной цели в водной среде. Детектировать объект можно, когда сигнал в 3 раза превышает вариацию фона (), на основании этого рассчитали максимальную дальность обнаружения, которая составила 30 м. Проведённая оценка расширяет область безопасного маневрирования автономных подводных аппаратов, а также при сближении и стыковке.

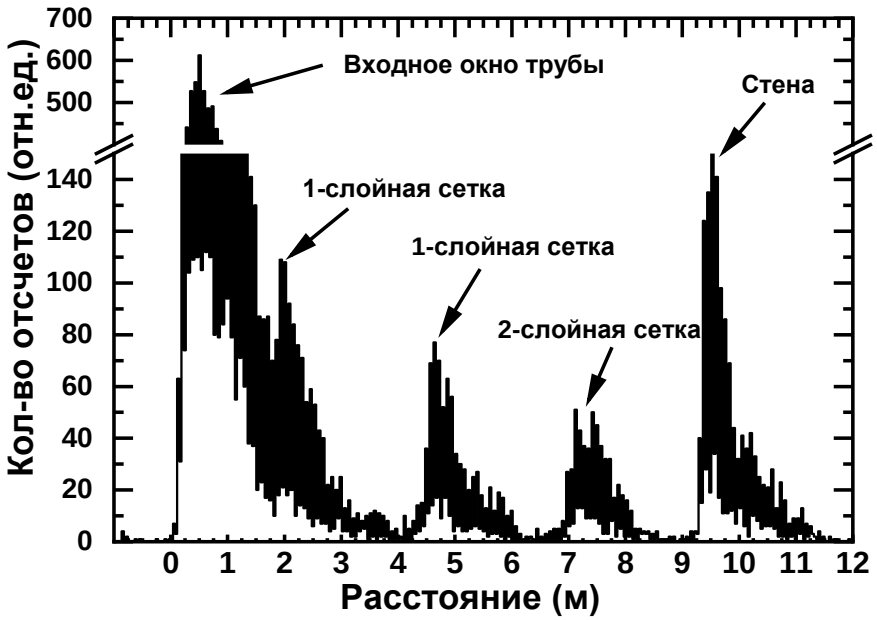

Рисунок 13 Гистограмма распределения обратно рассеянных фотонов по времени задержки (расстояния) измеренная лидаром при зондировании через 9 м толщи воды с установленными рассеивающими сетками на расстоянии 2.2 м, 4.5 и 7.2 м от входного окна.

Отметим, что в естественных акваториях дальность видимости кратно сокращается из-за присутствия микроорганизмов и органических пигментов в

большей степени в пресноводных [90], чем в океанах [91, 92, 93, 94]. Так, комплексный эксперимент [91] по измерению дальности видимости в Тихом океане у берегов Калифорнии и на Гавайях (США) показал, что дальность видимости под водой вдоль горизонтальной трассы на глубине 10, 20 и 30 метров до чёрной мишени диаметром 0.2 м не превышает 18 м.

Из Рисунок 13 следует, что вероятностный, сугубо статистический режим работы лидара, описанный в параграфе 2.1, обеспечивает зондирование многослойных облаков или последовательно расположенных рассеивающих объектов, например, сеток или оптически неоднородных слоёв, шлейфов стратификации.

Принимая во внимание, что зондировании подводной трассы проводилось импульсами с энергией 1-2 мкДж, характерной для диодных импульсных лазеров, открывается возможность создания лидаров на диодных лазерах с длиной волны генерации в области максимальной прозрачности воды 450-480 нм [80].

Таким образом, впервые, насколько нам известно, проведено зондирование объекта-стены на двойном проходе через слой воды толщиной 9 м импульсным лидаром на второй (532 нм) гармонике Nd:YAG-лазера (3 нс, 2 мкДж) с приёмником на стробируемом однофотонном лавинном фотодиоде (SPAD). Показано, что большое значение отношения сигнал/шум=35 от мишени-стены даёт оценку увеличения длины зондирования до 30 м с вероятностью ошибки обнаружения ~2% (при выполнении критерия 3σ) на двойном проходе, несмотря на предельно малую, безопасную для глаз ~1 мкДж/см^2, плотность энергии импульса лазера, а также дополнительные потери на трёх сетках-рассеивателях, последовательно установленных вдоль трассы.

4.2 Лазерное зондирование многослойных туманов

Маневрирование в сложных метеорологических условиях (плотные туманы, метель, шлейфы пожаров и др.) или в искусственно созданных оборонительных дымовых завес является высокорискованным действием, и, как правило,

принимается решение прекратить какое-либо движения до момента улучшения видимости. Поэтому разработка методов для визуализации препятствий в средах высокой оптической плотности и неоднородностей («увидеть сквозь туман») является высоко востребованным. Также отметим, что классические современные лидарные комплексы с фотоэлектронным детекторами не позволяют получать информации об внутренней структуре облаков высокой плотности (кучевые, грозовые).

Разработанный статистический лидар может позволяет «заглянуть внутрь» туманов и шлейфов высокой плотности, так как цифровой принцип работы, основанный на регистрации баллистических фотонов, позволяет регистрировать сигналы рассеяния от объектов, расположенных за оптически плотной преградой, т.е. регистрировать баллистические фотоны с вероятностью, а 10^-5 и менее. Для проверки предложенного зондирования сквозь оптически плотные среды была предпринята попытка исследовать структуру многослойных туманов и объектов за туманом. Зондирование аэрозолей проводили в наклонном тоннеле сцинтилляционного телескопа Баксанской Нейтринной Обсерватории, поскольку в данном месте формируются туманы высокой плотности вследствие взаимодействия воздушных масс разной температуры и влажности [95]. Лидар был установлен в туннеле, где формировались туманы высокой плотности. Трасса зондирования проходила в верхней части тоннеля, пучок лидара направляли на экран, установленный в 47 м от прибора (Рисунок 14). Была получена гистограмму обратного рассеяния при действии 10^5 лазерных импульсов, что соответствовало времени измерения в 20 секунд.

Были определены сигналы (Рисунок 14в):

1) Сигнал обратного рассеяния определили как интеграл фотоотсчетов на гистограмме от 0 до 45 м
2) Сигнал пропускания трассы определили как интеграл фотоотсчетов на гистограмме от 47 до 55 м

В этом тоннеле наблюдаются уникальные условия для тестирования нового макета лидара. Так в момент открытия ворот вспомогательного тоннеля БНО для перевозки сотрудников к лабораториям и нейтринным телескопам на электровозе, тёплый (16 - 20 ^{0}C) воздух извне поступает из-за принудительной вентиляции в наклонный тоннель, который наполнен влажным воздухом с более низкой температурой. При взаимодействии этих воздушных масс происходит конденсация паров воды. Фронт паров перемещается вверх по потоку приточной вентиляции с разной скоростью и прерываниями в разные дни. В нерабочие дни недели ворота не открываются, температура в наклонном тоннеле стабилизируется в отсутствии поступления тёплого воздуха, и атмосфера просветляется. Так в ходе эксперимента лидар успешно регистрировал изменение неоднородных туманов в наклонном тоннеле Рисунок 15.

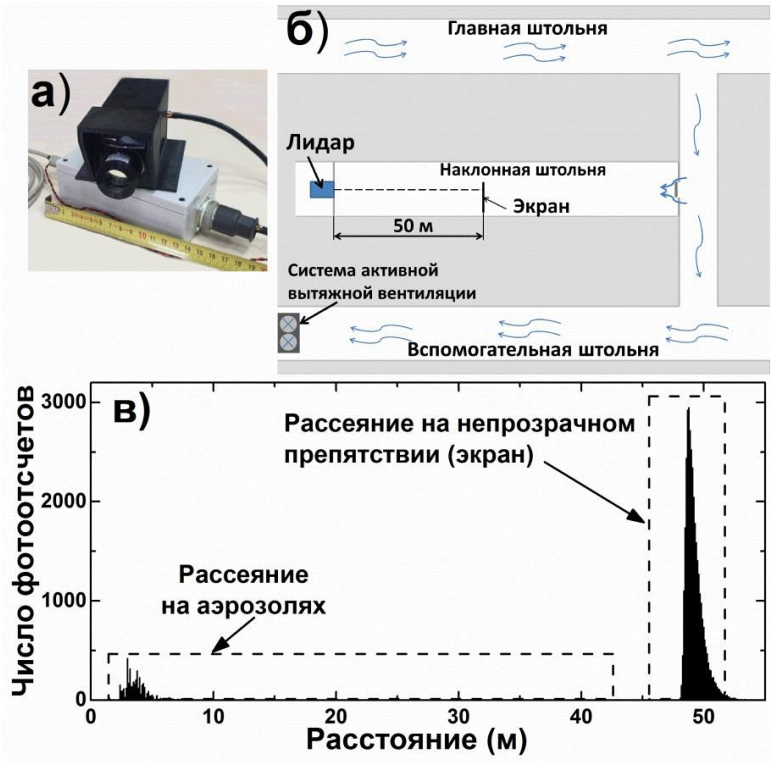

Рисунок 14 а) фото новой версии приёмо-передающего блока (черного цвета) лидара и интерфейсный блок (светлого фона); б) - схема тоннелей на участке БНО около сцинтилляционного телескопа; в) – типичный сигнал профиля обратного рассеяния в атмосфере, прилегающей к лидару, и сигнал от мишени на расстоянии 47 м.

Длительные лидарные измерения на протяжении трех дней позволили провести оценку динамики формирования и рассеяния плотных туманов в тоннеле (Рисунок 15). Показано, что при контакте с внешней атмосферой в течение двух часов сформировался плотный туман, для которого вероятность зарегистрировать фотон, прошедший через него до мишени и обратно, составляет менее 20 %. Плавное заполнение объёма тоннеля туманом наблюдалось в течение следующих двух дней.

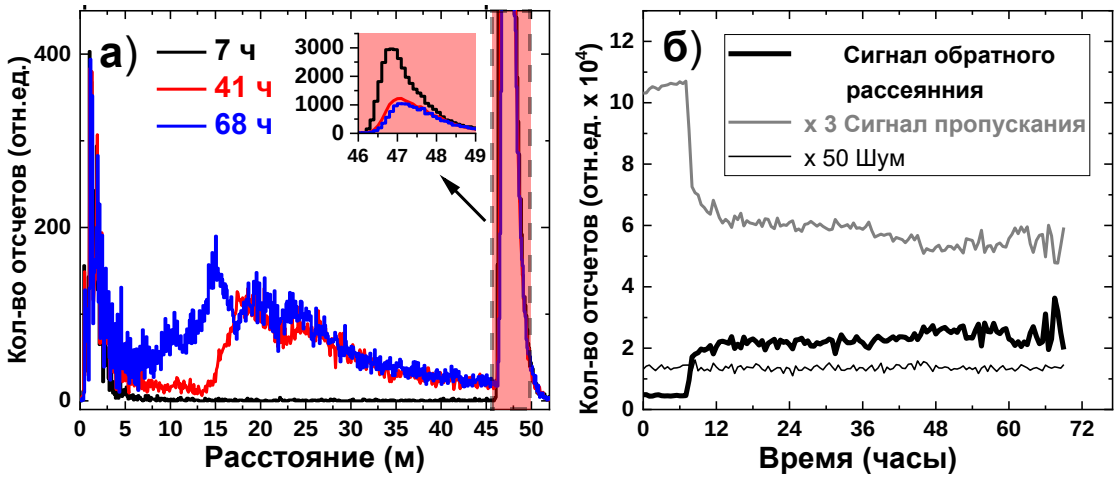

Рисунок 15 Лазерное зондирование многослойных туманов. (б) – три гистограммы обратно рассеянных фотонов для различных профилей плотности многослойных туманов; (в) – вариации сигнала обратного рассеяния (толстая линия); сигнал пропускания трассы на круговом обходе (серая линия, x3 для визуализации) и шумовые отсчёты (тонкая линия, x50 для визуализации) при формировании многослойного тумана.

Также в ходе эксперимента были зарегистрированы слоистые структуры туманов, например, на Рисунок 16а представлена гистограмма обратного рассеяния от 15 июля 2019 года в 11:54, а на Рисунок 16б представлена гистограмма обратного рассеяния в момент, когда облако тумана полностью заволокло приёмную апертуру лидара. Важно отметить, что в последнем случае вероятность регистрации сигнала от мишени составила только 6% от общего числа стартов, в то время как при чистой атмосфере вероятность составляла около 35%.

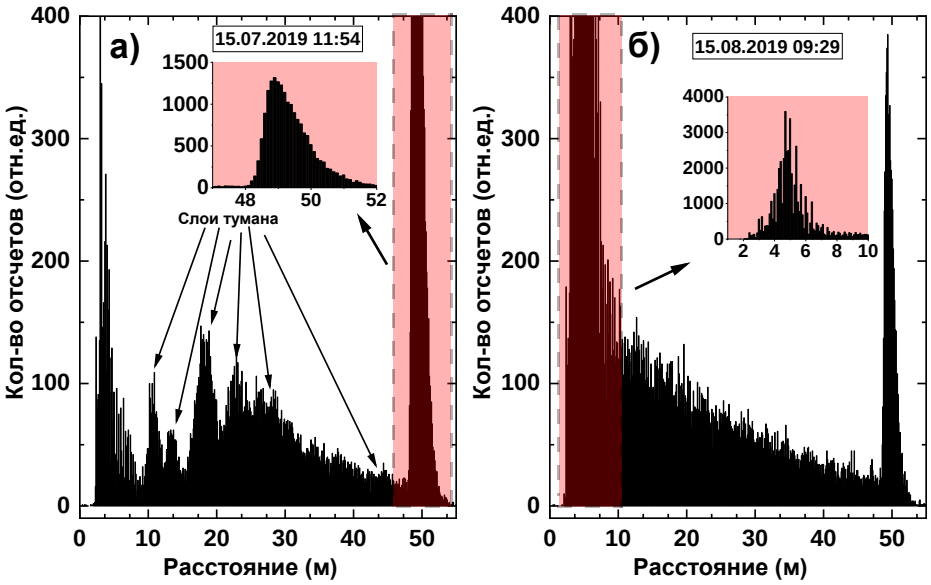

Рисунок 16 а) - пространственная модуляция коэффициента обратного рассеяния при образовании многослойного (5 – 6 слоев) тумана (15.07.2019, 11:24) с 3-кратным снижением коэффициента пропускания трассы; б) – зондирование экрана сквозь плотный туман с ~10-кратным увеличением коэффициента обратного рассеяния.

Глава 5. Зондирование аэрозолей тектонических аэрозолей

5.1 Зондирование тектонического аэрозоля вблизи вулкана Эльбрус

Как было подробно представлено во введении (раздел 1.4) лазерное дистанционное зондирование перспективно для мониторинга динамики эманации газов и аэрозолей в сейсмически активных областях [63]. Первые работы в это области проводили зондированием атмосферы вблизи поверхности вулканов и были обнаружено, что перед тектоническими событиями происходит увеличение скорости эмиссии различных газов, что связали с ростом внутренних напряжений в земной коре. Однако, результаты лазерного зондирования газов и аэрозолей не всегда коррелировали с последующими событиями (вулканическая активность, землетрясения), что было вызвано сильным влиянием открытой атмосферы (ветра, термические потоки), что приводило к маскированию тектонических газов. Поэтому, представляет интерес исследовать эманацию тектонических газов и аэрозолей в контролируемых условиях, когда влиянием погодных условий минимизировано. Для этого представляется подходящим проводить мониторинг вариации аэрозолей и газов в закрытом туннеле, расположенном вблизи сейсмически активного объекта (стык тектонических плит, вулкан). Туннели Баксанской нейтринной обсерватории Института физики Земли (ИФЗ) РАН хорошо соответствуют поставленным критериям: 1) подземный туннель, расположенный вблизи вулкана; 2) слабое влияние погодных условий (большая длина штольни); 3) непрерывная медленная откачка газов из туннеля, чтобы отслеживать вариацию динамики эманации выходящих тектонических газов, 4) наличие электропитания и доступа в сеть Интернет, для непрерывного сбора данных; 5) возможность оперативно организовать доступ к оборудованию в случае выхода его из строя.

Для поиска корреляции концентрации аэрозолей с сейсмической активностью был организован эксперимент над очагом вулкана Эльбрус. Аэрозольный лидар (см. раздел 3.2) был размещён в горячем (~40°C) тоннеле с

координатами 43°14’57.7”N, 42°43’19.5”E на высоте 1715 м над уровнем моря. Тоннель (длина ~50 м, диаметр около 3 м, с необработанными стенами и полом) ориентирован в направлении север-юг в конце главной штольни Баксанской нейтринной обсерватории (БНО) на удалении ~4000 м от входа. В тоннеле размещена низкофоновая геофизическая лаборатория института физики Земли (ИФЗ) РАН. Лидар разместили на плоской площадке, установленной на скальном основании у стены тоннеля невысоко от пола (~40 см) на удалении ~32 м от глухой стены. При этом до входа в тоннель оставалось около 20 м, чтобы снизить влияние вариаций аэрозолей у входа на результаты измерений. Трасса зондирования была наклонной с увеличением угла относительно пола. Пятно лазерного пучка размером 30х80 см на глухой стене было расположено на высоте 150 см от пола.

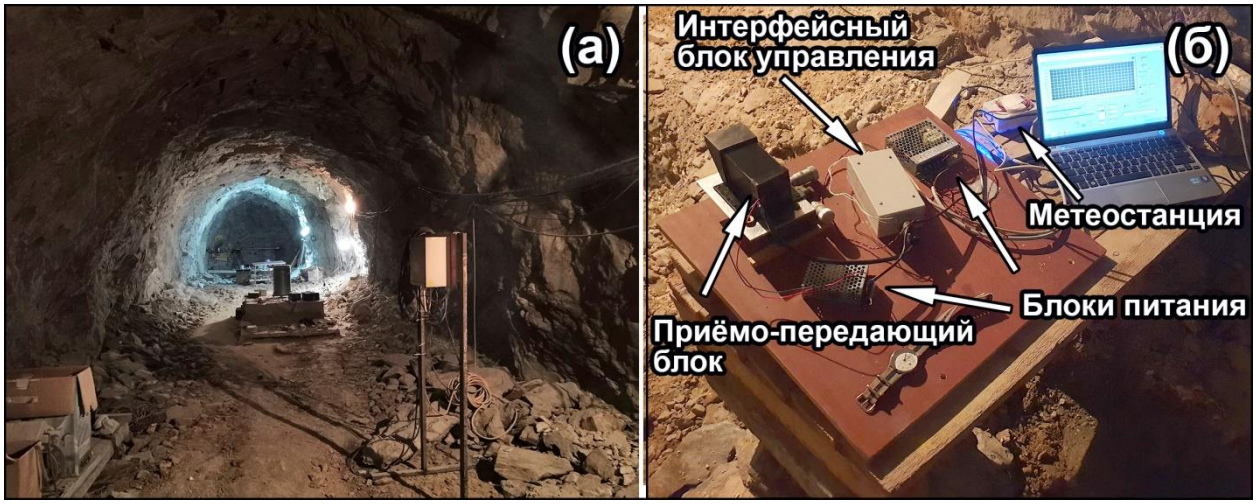

Рисунок 17 Горячий тоннель низкофоновой лаборатории ИФЗ (а) и платформа (б) с лидаром, интерфейсным блоком управления, метеостанцией и управляющим компьютером у правой стены тоннеля

Для измерения вариации аэрозолей лидар регистрировал гистограмму обратного рассеяния по 10^5 импульсам в течение 20 сек, а затем с заданным интервалом (10 мин) проводили следующее измерение и так далее. Далее по гистограмме проводилось интегрирование фотонов, рассеянных на аэрозолях и на мишени.

Результаты измерений представлены на Рисунок 18

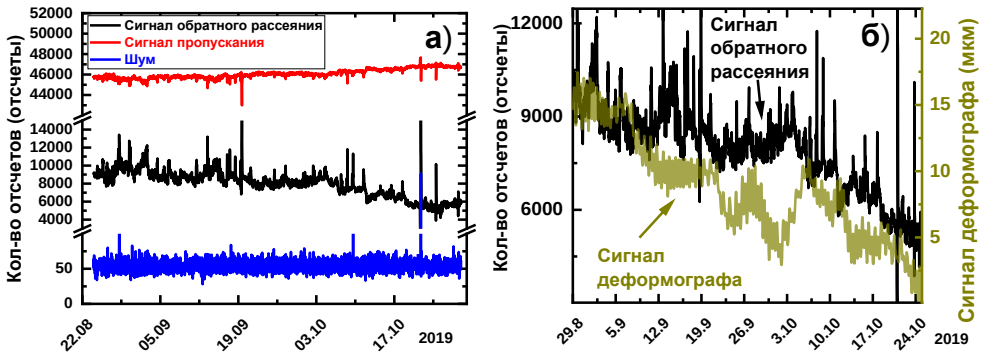

Рисунок 18 Результаты лидарного мониторинга вариации концентрации аэрозолей в тоннеле БНО над очагом вулкана Эльбрус. (а) - сезонное снижение коэффициента обратного рассеяния на аэрозолях (чёрная линия) с одновременным увеличением пропускания трассы лидарного зондирования (красная линия) и шумовой сигнал (нижняя синяя линия) с выбросами аэрозолей в объём тоннеля из трещин и пор; (б) – вариация сигнала обратного рассеяния (черная линия) и сигнала лазерного деформографа, установленного в БНО для непрерывного мониторинга деформации коры Земли.

По графикам видно устойчивое и заметное (двукратное) снижение аэрозолей (чёрная линия вверху), которое сопровождается повышением пропускания трассы (красная линия посередине). Сильная корреляция снижения выхода аэрозолей и увеличения пропускания трассы подтверждает, что наблюдаемое явление является процессом снижения выноса аэрозолей, а не следствием каких-либо технических неполадок. Впервые нам удалось зарегистрировать двукратное сезонное снижение выноса аэрозолей из вулканической камеры в объём тоннеля за два месяца (25.08 – 27.10.2019). При этом уровень шума был незначительным и постоянным. Подробнее этот эксперимент описан в статье [86].

Для автоматического удаленного сбора данных по мониторингу вариации аэрозолей, концентрации газов и метеопараметров в штольне Баксанской нейтринной обсерватории была разработана программа на языке программирования Python (Приложение 2). Программа состоит из нескольких модулей, каждый из которых отвечает за свои функции. Модуль bno_logger.py отвечает за журналирование работы программы. Модуль fig_ploting.py отвечает за

построение графиков по данным, полученным в ходе мониторинга. Модуль insert_sql_data.py отвечает за загрузку данных в удаленную базу данных SQL. Модули ldp_class.py, lidar_class.py и meteostation_class.py отвечают за взаимодействие с приборами лидар дифференциального поглощения разработанного в ИОФ РАН [96], аэрозольным лидаром разработанным автором и метеостанцией соответственно. Модуль read_sql_data.py отвечает за чтение данных мониторинга из удаленной базы данных SQL. В модуле sql_connect.py описана структура базы данных SQL.

Результаты аэрозольного мониторинга были сопоставлены с данными лазерного деформографа, который установлен в БНО для непрерывного мониторинга деформации коры Земли. Фурье-анализ данных лазерного деформографа и аэрозольного лидара (см. Рисунок 19) показал хорошее совпадение частоты суточных и полусуточных модуляций деформации коры Земли и вариаций выхода тектонических аэрозолей приливными волнами P_1K_1 и S_1. Приливные волны указаны согласно П. Мельхиору [97].

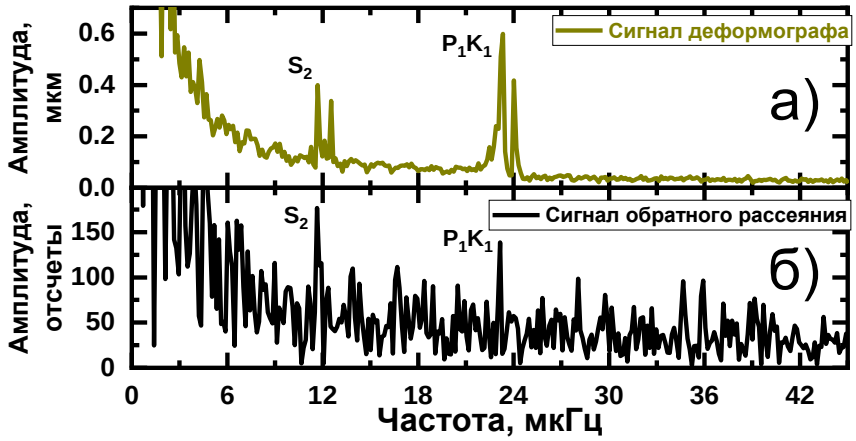

Рисунок 19 Фурье-спектр данных лазерного деформографа (а) и пропускания трассы лидарного зондирования (б), S_2, – полусуточные приливные волны связанные с Солнцем, P_1, K_1 – суммарный вклад двух компонент суточных колебаний связанных с Солнцем согласно П. Мельхиору [97]

Для подтверждения факта регистрации тектонических процессов с помощью лидара в режиме счета фотонов по вариации аэрозоля, была проведена оценка влияния метеопараметров на сигнал лидара. Спектральный Фурье-анализ данных за период с 23.08.2019 по 13.06.2020 показал, что в сигналах присутствует доминирующая частотная составляющая с периодом 24 часа и её вторая гармоника с периодом 12 часов. Подобная суточная и полусуточная модуляция сигнала лидара указывает на то, что лидар регистрирует как солнечные приливы, так и вклады флуктуаций температуры, давления и влажности, значения которых заметно меняются днём и ночью, но неодинаково по величине и по времени. Гармонический и регрессионный анализ данных лидара и метеопараметров показал, что основным фактором шума является вариация влажности атмосферного воздуха вне БНО. Анализ фазовых портретов сигналов обеспечил измерение запаздывания (~80 минут) перемещения по штольне на расстояние ~4000 м порции влажного воздуха в горячий тоннель и скорости потока вентиляции (~0.8 м/с), соответственно [98].

5.2 Лидарный мониторинг динамики аэрозолей, индуцированных аэроионами

Ранее (см. раздел 5.1) была обнаружена корреляция между лидарным сигналом рассеяния на аэрозолях и деформацией земной коры, измеряемой лазерным деформографом [86]. Однако, было обнаружено несколько случаев, когда были зарегистрированы флуктуации аэрозольного сигнала, но при этом отсутствовали изменения деформации земной коры, атмосферного давления и температуры. Было высказано предположение о том, что выход радона из скального основания приводит к увеличению концентрации аэроионов, которые являются центрами конденсации. Также отметим, что большие вариации концентрации аэроионов отражают сейсмическую активность, то есть являются потенциальными кандидатами на индикаторы будущих сейсмических событий [99]. Тектонические газы выделяются через сеть мелких трещин в породах, и их трудно обнаружить визуально вследствие очень слабых потоков. В случае установки датчиков вдали от мест дегазации, не получится зарегистрировать изменения в динамике эманации газов. Более того, сеть трещин в породах не является статичной и постоянно изменяется из-за вариации направления и силы процессов сжатия-растяжения в земной коре. Одним из способов решения этой проблемы является создание огромной сети датчиков аэроионов, однако и в этом случае сложно добиться воспроизводимых результатов, так ка сложно добиться требуемой высокой плотности распределения сенсоров. Альтернативным подходом является использование методов дистанционного зондирования, которые обеспечивают высокое пространственное разрешение на протяженных трассах, а также позволяет сканировать большие площади с высокой скоростью получения карт сигналов.

В связи с этим было проведено систематическое исследование влияния концентрации аэроионов на результаты аэрозольного зондирования безопасным для глаз лидаром, разработанным в ИОФ РАН (см. раздел 3.2). Первая цель исследования заключалась в количественной оценке того, как флуктуации концентрации аэроионов могут влиять на концентрацию аэрозолей, чтобы объяснить необычные всплески концентрации аэрозолей, обнаруженные нами ранее во время долгосрочного мониторинга в тоннеле рядом с вулканом Эльбрус.

Вторая цель исследования состояла в оценке возможности обнаружения малых флуктуаций аэрозолей, вызванных изменением концентрации аэроионов, то есть оценить перспективы применения аэрозольного лидара для количественного измерения концентрации аэроионов.

Для измерений был выбрана подвальная лаборатория в НИИЯФ МГУ, расположенный на глубине 12 метров под землей (55°41'56.1" с.ш., 37°32'28.2" в.д.), поскольку эта подземная лаборатория отвечала следующим требованиям: отсутствие солнечного света (предотвращение генерации аэроионов ультрафиолетовыми фотонами); наличие вентиляции для подачи внешнего атмосферного воздуха в место проведения эксперимента для модуляции концентрации аэроионов. Для независимого измерения концентрации аэроионов был использован счетчик аэроионов «Сапфир-3М», модифицированный сотрудниками ИЯИ РАН [100]. Данная модификация прибора оснащена подогревателем аспирационной камеры счётчика ионов для испарения влаги с поверхности изоляторов измерительных сеток, что повышает точность и стабильность относительных измерений плотности аэроионов обоих знаков в условиях повышенной влажности подземных помещений.

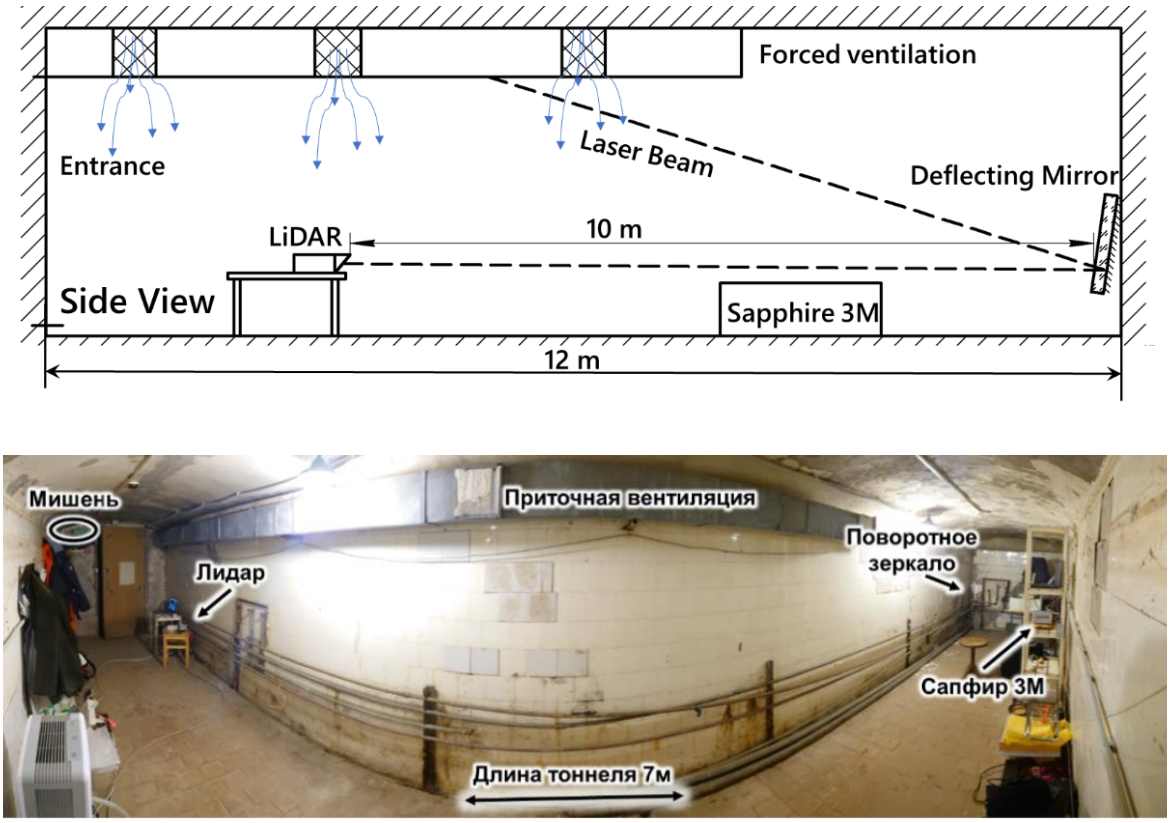

Рисунок 20 Схема эксперимента (сверху) и панорамная фотография (снизу), иллюстрирующая расположение экспериментальных установок в помещении (лидар установлен у левой стены около входа, поворотное зеркало установлено в конце тоннеля, счетчик аэроионов расположен у правой стены, а вход приточной вентиляции расположен у потолка на левой стене в середине помещения).

В результате лидарного измерения формируется гистограмма обратного рассеяния (см. Рисунок 21), на которой мы определили сигналы. Первый пик на гистограмме (от 0 до 8 м) обусловлен обратным рассеянием на аэрозолях (серый цвет), поэтому сумма фотоотсчетов на этом участке была принята как сигнал рассеяния на аэрозолях. Обратное рассеяние от зеркала (середина трассы зондирования, 10 м от лидара) не использовали, так как оно обусловлено отражением от поверхности поворотного зеркала, главным образом от осевшей пыли. Обратное рассеяние на аэрозолях между зеркалом и задней стеной, зарегистрированное лидаром, очень мало для достоверных измерений и не может быть принято в качестве сигнала. Третий пик на гистограмме на Рисунок 21

(красный цвет) соответствует рассеянию лазерного импульса на задней стене тоннеля, поэтому сумма фотоотсчетов на этом участке была определена как сигнал «пропускания трассы», то есть, прохождение лазерного излучения до стены и обратно. Этот сигнал пропорционален прозрачности воздуха – чем прозрачнее воздух в тоннеле, тем больше сигнал обратного рассеяния от стены.

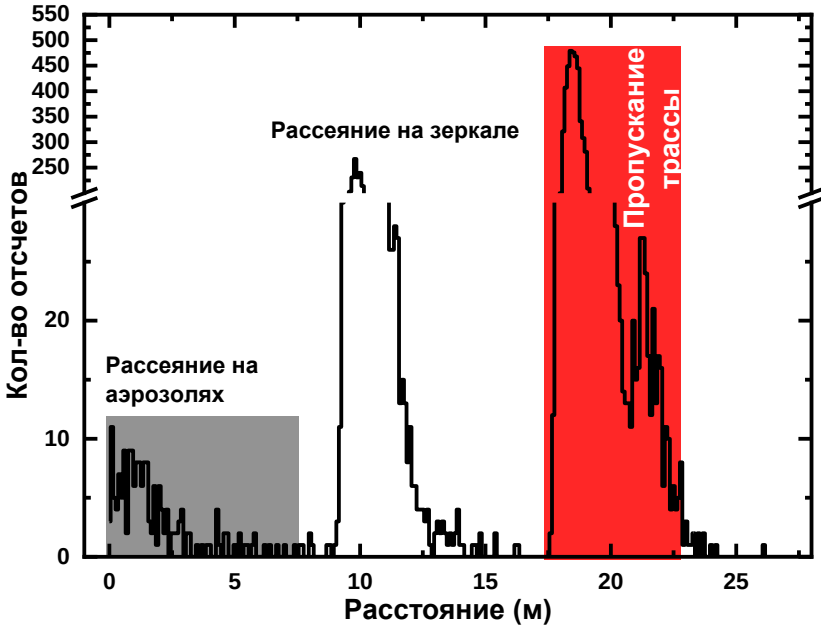

Рисунок 21 Гистограмма распределения фотонов по трассе зондирования. Сигнал аэрозольного рассеяния определили как сумму фотоотсчётов от фотонов, рассеянных на первых 8 м трассы (серый цвет). Пик в гистограмме на расстоянии 20 м от лидара соответствует рассеянию на стене, поэтому интеграл данного пика был принят в качестве сигнала пропускания трассы (красный цвет).

На Рисунок 22 представлены данные, полученные за 6 дней измерений (05-11.12.2022): концентрация аэрозолей в относительных единицах (лидар) и аэроионов (счетчик ионов “Сапфир-3М”), а также значения температуры, давления и относительной влажности воздуха. Периодическая модуляция сигналов обусловлена включением вентиляции в помещении. Сигнал рассеяния на аэрозолях (не показан на Рисунок 22) был очень сильно зашумлен, и, нам не удалось обнаружить его корреляции с другими измеряемыми параметрами. В случае малых концентраций аэрозолей целесообразно регистрировать сигнал прозрачности

атмосферы, который значительно более чувствителен к малым величинам вариации концентрации (см. Рисунок 22 а). Вариации концентрации ионов и влажности определяются включением/выключением вентиляции. Из Рисунок 22 (б) видно, что соотношение положительных и отрицательных ионов в воздухе не является постоянным при работающей вентиляции. Для количественного описания содержания ионов в воздухе часто используют отношение концентраций положительных и отрицательных ионов («коэффициент униполярности»). Вентиляция оказывала меньшее влияние на концентрацию положительных ионов в воздухе по сравнению с отрицательными, поэтому коэффициент униполярности увеличивался при включенной вентиляции, а при выключенной – уменьшался. Поскольку при измерениях температура воздуха и атмосферное давление были практически постоянными (±0.1°C, ±5 мм. рт. ст.), то обнаруженная корреляция обусловлена гидратацией аэроионов. Отметим, что сигнал пропускания трассы (прозрачность воздуха) запаздывает по сравнению с началом увеличения концентрации ионов. Ионы, появляющиеся в воздухе в результате радиоактивного распада радона, становятся центрами конденсации для водяных паров и формируют взвешенные в воздухе микроскопические капли, которые и детектирует аэрозольный лидар.

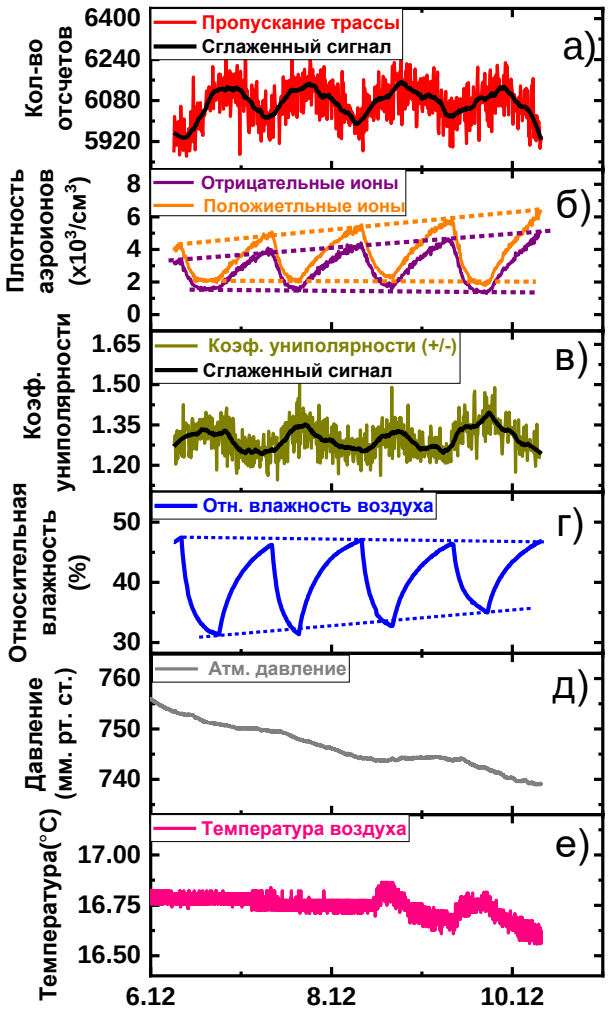

Рисунок 22 Лидарный мониторинг аэрозолей, индуцированных аэроионами: (а) – лидарный сигнал прозрачности атмосферы (красный цвет) и отфильтрованный сигнал (черный цвет); (б) – концентрация положительных (оранжевый цвет) и отрицательных (фиолетовый цвет) аэроионов; (в) – коэффициент униполярности (табачный цвет) и его усредненное значение (черный цвет); (г) – относительная влажность воздуха (датчик Honeywell HIH-4000-003, ±3.5%RH); (д) – давление воздуха (датчик MPXA4100A6U/T1, ±11.25 мм. рт. ст.); (е) –температура воздуха (датчик AD22100K, ±0.5 °C).

Сопоставление сигналов аэрозольного рассеяния, концентраций ионов и коэффициента униполярности на Рисунок 23 выявило высокие значения коэффициента корреляции Спирмана (ρ=0.84). Следует отметить, что на некоторых участках графика на Рисунок 22 видна линейная зависимость между

коэффициентом униполярности и лидарным сигналом (см. левую часть на графике с корреляцией, близкой к единице ρ=0.9997). Эти точки данных соответствуют моменту отключения вентиляции, что приводит к увеличению концентрации радона, а, соответственно, концентрация аэроионов растет, что приводит к образованию водного аэрозоля (Рисунок 24).

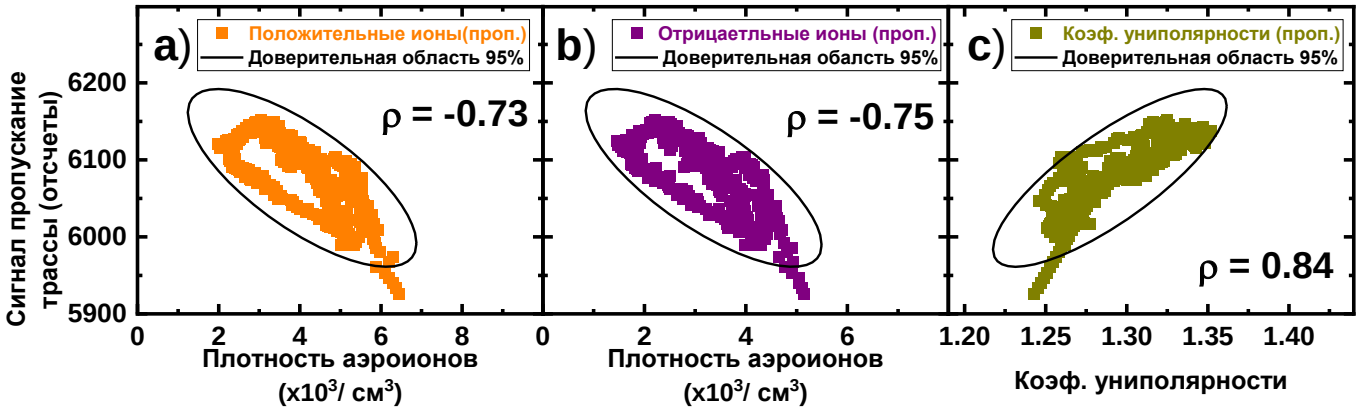

Рисунок 23. Графики корреляции лидарного сигнала и концентраций аэроионов: (а) – концентрация положительных ионов от сигнала прозрачности атмосферы (оранжевый цвет); (б) – концентрация отрицательных ионов от сигнала прозрачности атмосферы (фиолетовый цвет); (в) – коэффициент униполярности в зависимости от сигнала прозрачности атмосферы (коричневый цвет). Значения коэффициента корреляции Спирмана (ρ) представлены на рисунке.

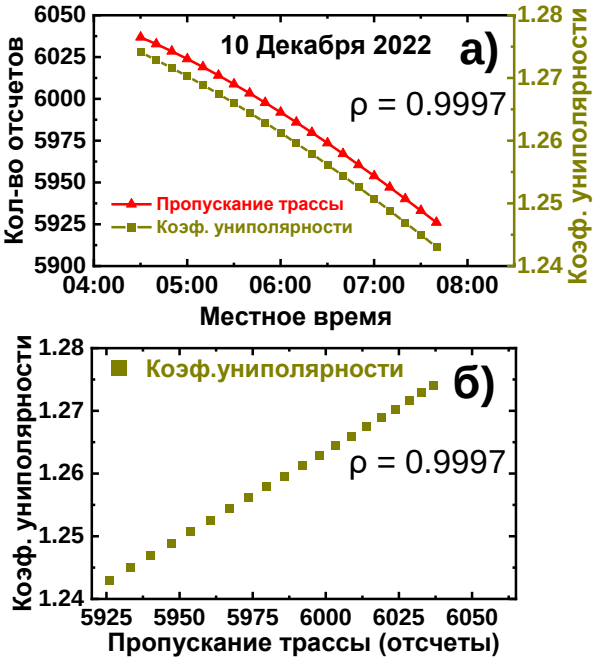

Рисунок 24 (а) – временной ряд данных лидарного сигнала прозрачности атмосферы и коэффициента униполярности. (б) – График корреляции коэффициента униполярности и сигнала прозрачности атмосферы (лидарный сигнал).

Таким образом, сильная корреляция коэффициента униполярности и прозрачности атмосферы открывает перспективы применения лидара для непрерывного мониторинга соотношения аэроионов. Особый интерес здесь представляет возможность мониторинга тектонических событий. Поскольку методы лазерного дистанционного зондирования позволяют регистрировать сигналы на длинных трассах при увеличении числа проходов с высоким пространственным и временным разрешением, то можно значительно повысить чувствительность измерений вариации аэрозолей при накоплении деформаций в земной коре перед тектоническими событиями.

В качестве продолжения работы по поиску корреляций с сейсмическими событиями было принято решение установить наиболее продвинутую версию аэрозольного лидара (двухканальный лидар) в районе с высокой сейсмической активностью. Для этого было выбрано помещение Института вулканологии и

сейсмологии ДВО РАН (ИВиС ДВО РАН) на п-ове Камчатка. Так как количество сейсмических событий на п-ове Камчатка значительно больше, чем в тоннеле БНО. Первоначально лидар был установлен на сейсмической станции «Петропавловск» (г. Петропавловск-Камчатский, ул. Гагарина, д. 81), однако, при пробных измерениях сигналов в течение суток обнаружили, что высокая влажность приводит к конденсации водяных паров на оптических элементах лидара и на зеркалах (для увеличения чувствительности измерений в ограниченном пространстве увеличили трассу зондирования с помощью трех зеркал). Было принято решение транспортировать оборудование в подвал ИВиС, где расположена скважина, пробуренная в скальном основании. Непосредственно рядом со скважиной (Рисунок 25) установили измеритель концентрации аэроионов Сапфир-3М, доработанный в ИЯИ РАН. В том же помещении установили двухканальный аэрозольный лидар. Для увеличения чувствительности измерений аэрозолей было необходимо увеличить трассу зондирования, для чего использовали три алюминиевых зеркала. Также в помещении была установлена метеостанция для измерения температуры, влажности воздуха, давления и концентрации углекислого газа. Управление приборами и автоматизированный сбор данных осуществляли с помощью компьютера, к которому был обеспечен удалённый доступ через сеть Интернет.

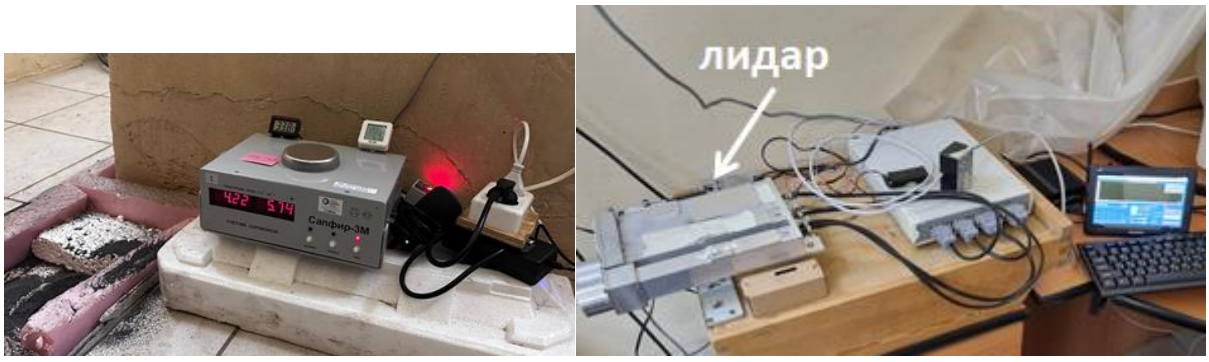

Рисунок 25 Фотография счетчика аэроионов Сапфир-3М (слева), установленного вблизи выхода скважины в подвале ИВиС ДВО РАН. Фотография двухканального аэрозольного лидара (справа), установленного в том же подвальном помещении.

На Рисунок 26 представлены временные ряды различных сигналов аэрозольного лидара и концентрации аэроионов. Рост концентрации аэроионов в начале измерений связан с тем, что при установке оборудования работала вентиляция в помещении, и при ее отключении (после запуска в работу всех приборов) происходил медленный рост концентрации аэроионов. Лидарные сигналы (сигнал рассеяния на аэрозоле и сигнал прозрачности атмосферы) были достаточно стабильными, и нам не удалось зарегистрировать существенных вариаций сигналов за прошедший период измерений. Детальный анализ показал, что коэффициенты корреляции между лидарными сигналами и концентрацией ионов, и коэффициентом униполярности были достаточно малыми (см. Рисунок 27). В то же время, если построить корреляционные графики для данных начального периода, когда концентрации аэроионов сильно варьируются, то можно обнаружить корреляции с сигналами лидара. Корреляционные зависимости аналогичны тем, что были представлены выше (Рисунок 23), однако были менее ярко выражены, что обусловлено меньшим диапазоном вариации концентрации аэроионов.

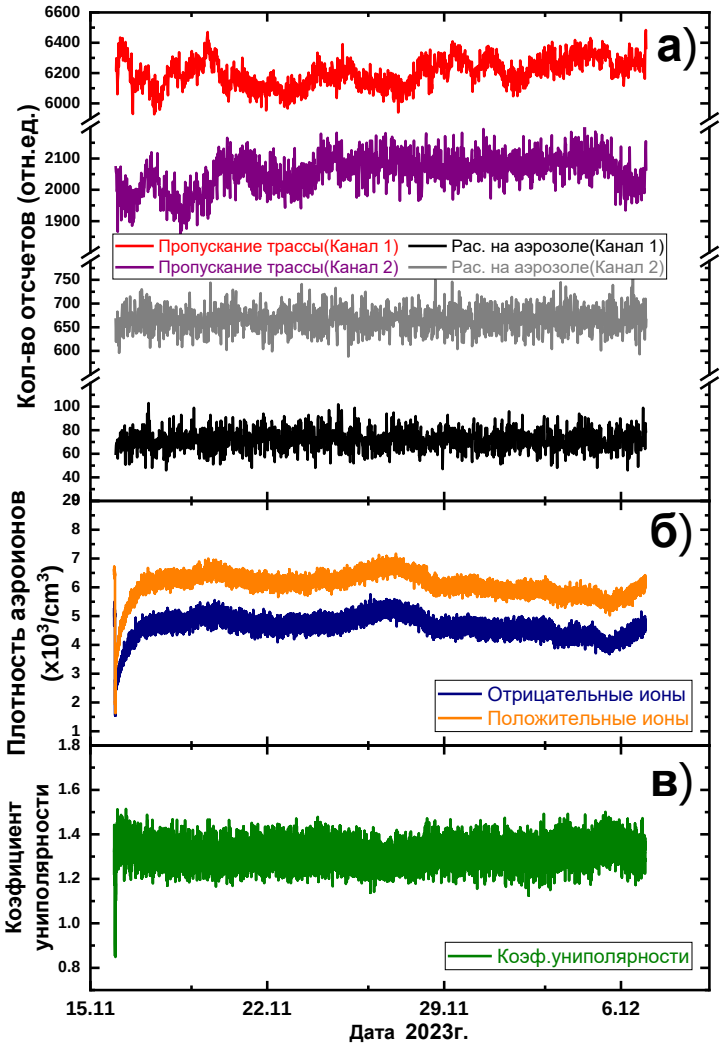

Рисунок 26 Сигналы аэрозольного зондирования в подвале ИВиС ДВО РАН: (а) сигнал пропускания трассы зондирования (красный и фиолетовый цвета) и сигналы рассеяния на аэрозолях (серый и черный) 1 и 2 каналах аэрозольного лидара; (б) концентрации положительных (оранжевый цвет) и отрицательных (синий цвет) аэроионов; (в) коэффициент униполярности (зеленый цвет).

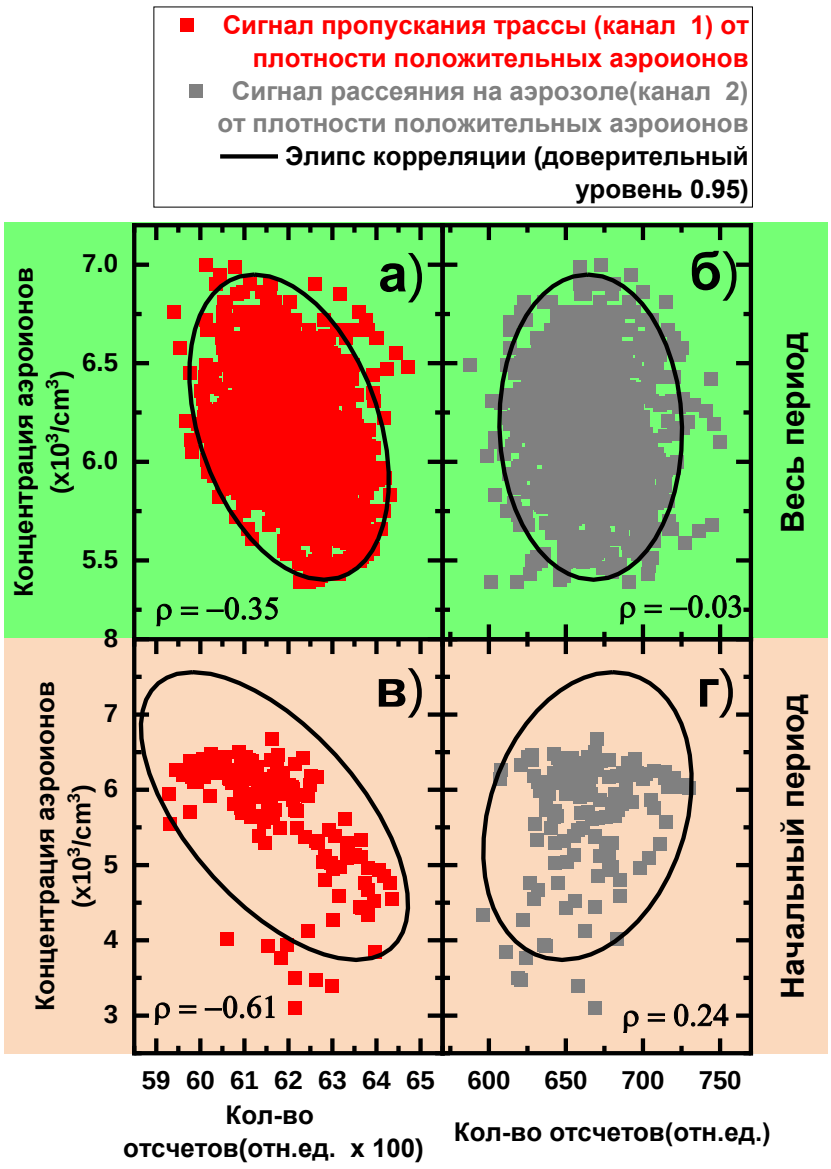

Рисунок 27 Графики корреляций плотности положительных ионов и лидарных сигналов (сигнал оптического пропускания атмосферы, сигнал рассеяния на аэрозолях). (а) и (б) – корреляции концентрации положительных аэроионов, сигнала пропускания трассы (красный цвет) и сигнала рассеяния на аэрозоле (серый цвет) соответственно для всей выборки данных (салатовый фон). (в) и (г) – корреляции концентрации положительных аэроионов, сигнала пропускания трассы (красный цвет) и сигнала рассеяния на аэрозоле (серый цвет) соответственно для начального периода, соответствующего значительной вариации концентрации аэроионов

Заключение

На основе результатов проведенных исследований сформулированы основные выводы диссертационной работы:

1. Показано и экспериментально обосновано, что при накачке инжекционного диодного лазера током разряда ёмкости, который коммутируется биполярным транзистором в лавинном режиме работы, генерируется лазерный импульс с пикосекундной ступенькой на фронте

2. Предложен адаптационный режим работы лидара и достигнута сантиметровая точность дальнометрии в лидаре с новым подходом измерения расстояния. Улучшение достигнуто за счёт использования диодного лазера с наносекундным импульсом с ступенькой на фронте в цифровом лидаре с лавинным фотодиодом и разрешением по времени равным 250 пс.

3. Продемонстрировано, что основанный на регистрации баллистических фотонов цифровой принцип работы микроджоульного лидара позволяет обнаруживать объекты сквозь оптические плотные среды (плотные туманы и шлейфы дымов, зеркальные поверхности, и др.). Впервые прямым измерением микроджоульным лидаром была исследована многослойная структура тумана.

4. Впервые обнаружена модуляция лидарного аэрозольного эхо-сигнала тектоническими газами, эмиссия которых вызвана деформацией коры Земли приливными волнами.

5. Впервые прямым измерением оптической прозрачности атмосферы в закрытом объёме подземной лаборатории обнаружена динамика индуцированной ионами генерации аэрозолей.

Список используемых сокращений

LiDAR — обнаружение и определение дальности с помощью света (англ. Light Detection And Ranging),

TDC — временно-цифровой преобразователь (англ. Time to Digital Converter),

ФЭУ — фотоэлектронный умножитель,

ЛФД — лавинный фотодиод (англ. Avalanche Photodiode, APD),

TCSPC — коррелированный по времени счет одиночных фотонов (англ. Time-Correlated Single Photon Counting),

DIAL — лидар дифференциального поглощения (англ. Differential Absorption Lidar),

FWHM — полная ширина на уровне половины высоты (англ. Full Width at Half Maximum),

SPAD — однофотонный лавинный фотодиод (англ. Single Photon Avalanche Diode),

БНО — Баксанская Нейтринная Обсерватория,

ИФЗ — Институт физики Земли им. О. Ю. Шмидта РАН,

ИВиС ДВО РАН — Институт вулканологии и сейсмологии ДВО РАН на полуострове Камчатка.

Библиографический список использованной литературы

1. Першин, С. М. Лидар / С. М. Першин // Большая российская энциклопедия. – Москва, 2010. – Т. 17. – С. 451-452.

2. Spaceborn laser altimeter based on the single photon diode receiver and semiconductor laser transmitter / S. Pershin, V. Linkin, V. Makarov [et al.] // Conference on Lasers and Electro-Optics / Citation Key: Pershin:91. – Optica Publishing Group, 1991. – P. CFI1.

3. Mars Polar Lander. – URL: https://nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=1999-001A (дата обращения: 01.01.2024). – Текст : электронный.

4. Концепция инфракрасного дистанционного газоанализатора лидарного типа для мониторинга антропогенных загрязнений / В. В. Мещеринов, М. В. Спиридонов, В. А. Казаков, А. В. Родин // Квантовая электроника. – 2020. – Т. 50. – № 11. – С. 1055-1062.

5. Измерения содержания аэрозолей в тропосфере Арктического региона дистанционными и контактными методами с борта самолеталаборатории Як-42Д" Росгидромет" / А. Кузьмичев, Т. Бабухина, А. Ганьшин [и др.] // Метеорология и гидрология. – 2016. – № 5. – С. 88-97.

6. Radon and metallic aerosols emanation before strong earthquakes and their role in atmosphere and ionosphere modification / S. A. Pulinets, V. A. Alekseev, A. D. Legen’ka, V. V. Khegai // Middle Atmosphere: Changes and Electrodynamics. – 1997. – Vol. 20. – № 11. – P. 2173-2176.

7. Межерис, Р. М. Лазерное дистанционное зондирование / Р. М. Межерис; ред. А. Б. Карасева; пер. В. И. Городецкий, В. В. Филюшкина. – Москва : Мир, 1987. – 550 с.

8. Collis, R. T. H. Lidar measurement of particles and gases by elastic backscattering and differential absorption / R. T. H. Collis, P. B. Russell // Laser Monitoring of the Atmosphere / ed. E. D. Hinkley. – Berlin, Heidelberg : Springer Berlin Heidelberg, 1976. – P. 71-151.

9. Grant, W. B. Optical Remote Measurement of Toxic Gases / W. B. Grant, R. H. Kagann, W. A. McClenny // Journal of the Air & Waste Management Association. – 1992. – Vol. 42. – № 1. – P. 18-30.

10. Pal, S. Monitoring Depth of Shallow Atmospheric Boundary Layer to Complement LiDAR Measurements Affected by Partial Overlap / S. Pal // Remote Sensing. – 2014. – Vol. 6. – № 9. – P. 8468-8493.

11. Sathe, A. A review of turbulence measurements using ground-based wind lidars / A. Sathe, J. Mann // Atmos. Meas. Tech. – 2013. – Vol. 6. – № 11. – P. 3147-3167.

12. Лидарные технологии дистанционного зондирования параметров атмосферы / В. Д. БУРЛАКОВ, С. И. ДОЛГИЙ, А. П. МАКЕЕВ [и др.] // Оптика атмосферы и океана. – 2013. – Т. 26. – № 10. – С. 829-837.

13. William B. Grant. Differential absorption and Raman lidar for water vapor profile measurements: a review / William B. Grant // Optical Engineering. – 1991. – Vol. 30. – № 1. – P. 40-48.

14. Pershin, S. M. New generation of the portable backscatter lidar with eye-safe energy level for environmental sensing / S. M. Pershin // Atmospheric Propagation and Remote Sensing III. – SPIE, 1994. – Vol. 2222. – P. 392-401.

15. Бухарин, А. В. Теоретическое рассмотрение лидара обратного рассеяния с безопасным для глаз уровнем излучения / А. В. Бухарин, С. М. Першин // Оптика атмосферы и океана. – 1994. – Т. 7. – № 4. – С. 521-537.

16. Першин, С. ЛИДАР НОВОГО ПОКОЛЕНИЯ ДЛЯ СБЛИЖЕНИЯ, ПРИЧАЛИВАНИЯ, СТЫКОВКИ КОСМИЧЕСКИХ АППАРАТОВ И НАВИГАЦИИ БЕСПИЛОТНЫХ ПЛАТФОРМ / С. Першин // Новые материалы и технологии для ракетно-космической и авиационной техники. – 2017. – С. 86-103.

17. Royo, S. An Overview of Lidar Imaging Systems for Autonomous Vehicles / S. Royo, M. Ballesta-Garcia // Applied Sciences. – 2019. – Т. 9. – № 19.

18. Debeunne, C. A Review of Visual-LiDAR Fusion based Simultaneous Localization and Mapping / C. Debeunne, D. Vivet // Sensors. – 2020. – Т. 20. – № 7.

19. Three-dimensional laser radar with APD arrays / Rick Heinrichs, Brian F. Aull, Richard M. Marino [et al.] // Proc.SPIE. – 2001. – Vol. 4377. – P. 106-117.

20. Marino, R. M. Jigsaw: a foliage-penetrating 3D imaging laser radar system / R. M. Marino, W. R. Davis // Lincoln Laboratory Journal. – 2005. – Vol. 15. – № 1. – P. 23-36.

21. LiDAR Velodyne HDL-64e calibration using pattern planes / G. Atanacio, J.-J. Gonzalez-Barbosa, J. Hurtado [и др.] // International Journal of Advanced Robotic Systems. – 2011. – Т. 8.

22. Velodyne VLS-128. – URL: https://data.ouster.io/downloads/velodyne/user-manual/alpha-prime-rev4-user-manual.pdf (date accessed: 01.01.2024). – Text : electronic.

23. Velodyne VLP-16. – URL: https://data.ouster.io/downloads/velodyne/user-manual/vlp-16-user-manual-revf.pdf (дата обращения: 01.01.2024). – Текст : электронный.

24. M. Bijelic. A Benchmark for Lidar Sensors in Fog: Is Detection Breaking Down? / M. Bijelic, T. Gruber, W. Ritter // 2018 IEEE Intelligent Vehicles Symposium (IV) 2018 IEEE Intelligent Vehicles Symposium (IV) / journalAbbreviation: 2018 IEEE Intelligent Vehicles Symposium (IV). – 2018. – P. 760-767.

25. Басов, Н. Г. Полупроводниковые лазеры / Н. Г. Басов, П. Г. Елисеев, Ю. М. Попов // Успехи физических наук. – 1986. – Т. 148. – № 1. – С. 35-53.

26. Звелто, О. Принципы лазеров / О. Звелто. – 4. – СПб : Издательство «Лань», 2008. – 720 с.

27. Дьяконов, В. Лавинные транзисторы и тиристоры. Теория и применение / В. Дьяконов. – Солон-Пресс, 2008. – 382 с.

28. Agrawal, G. P. Semiconductor lasers / G. P. Agrawal, N. K. Dutta. – Springer Science & Business Media, 2013.

29. Тарасов, И. С. Мощные полупроводниковые лазеры на основе гетероструктур раздельного ограничения / И. С. Тарасов // Квантовая электроника. – 2010. – Т. 40. – № 8. – С. 661-681.

30. Lau, K. Y. Gain switching of semiconductor injection lasers / K. Y. Lau // Applied Physics Letters. – 1988. – Vol. 52. – № 4. – P. 257-259.

31. K. Y. Lau. Short-pulse and high-frequency signal generation in semiconductor lasers / K. Y. Lau // Journal of Lightwave Technology. – 1989. – Vol. 7. – № 2. – P. 400-419.

32. Aspin, G. J. The effect of cavity length on picosecond pulse generation with highly rf modulated AlGaAs double heterostructure lasers / G. J. Aspin, J. E. Carroll, R. G. Plumb // Applied Physics Letters. – 1981. – Vol. 39. – № 11. – P. 860-861.

33. Picosecond optical pulse generation by impulse train current modulation of a semiconductor laser / R. A. Elliott, H. DeXiu, R. K. DeFreez [et al.] // Applied Physics Letters. – 1983. – Vol. 42. – № 12. – P. 1012-1014.

34. Gain-switched picosecond pulse (<10 ps) generation from 1.3 mu m InGaAsP laser diodes / H. . -F. Liu, M. Fukazawa, Y. Kawai, T. Kamiya // IEEE Journal of Quantum Electronics. – 1989. – Vol. 25. – № 6. – P. 1417-1425.

35. Observation of ultrashort (<4 ps) gain-switched optical pulses from long-wavelength multiple quantum well lasers / R. Nagarajan, T. Kamiya, A. Kasukawa, H. Okamoto // Applied Physics Letters. – 1989. – Vol. 55. – № 13. – P. 1273-1275.

36. Sub-5-ps optical pulse generation from a 1.55-μm distributed-feedback laser diode with nanosecond electric pulse excitation and spectral filtering / S. Chen, A. Sato, T. Ito [et al.] // Optics Express. – 2012. – Vol. 20. – № 22. – P. 24843-24849.

37. 7-ps optical pulse generation from a 1064-nm gain-switched laser diode and its application for two-photon microscopy / Y. Kusama, Y. Tanushi, M. Yokoyama [et al.] // Optics Express. – 2014. – Vol. 22. – № 5. – P. 5746-5753.

38. Two-photon bioimaging with picosecond optical pulses from a semiconductor laser / H. Yokoyama, H. Guo, T. Yoda [et al.] // Optics Express. – 2006. – Vol. 14. – № 8. – P. 3467-3471.

39. Kilpelä, A. Laser pulser for a time-of-flight laser radar / A. Kilpelä, J. Kostamovaara // Review of Scientific Instruments. – 1997. – Vol. 68. – № 6. – P. 2253-2258.

40. Dyakonov, M. Theory of streamer discharge in semiconductors / M. Dyakonov, V. Yu, A. Kachorovskii // Soviet Physics JETP. – 1985. – Vol. 61. – P. 1125.

41. Multistreamer regime of GaAs thyristor switching / S. N. Vainshtein, A. J. Kilpela, J. T. Kostamovaara [et al.] // IEEE Transactions on Electron Devices. – 1994. – Vol. 41. – № 8. – P. 1444-1450.

42. Superfast fronts of impact ionization in initially unbiased layered semiconductor structures / P. Rodin, U. Ebert, W. Hundsdorfer, I. V. Grekhov // Journal of Applied Physics. – 2002. – Vol. 92. – № 4. – P. 1971-1980.

43. Detectors for Photon Counting // Advanced Time-Correlated Single Photon Counting Techniques / eds. W. Becker [et al.]. – Berlin, Heidelberg : Springer Berlin Heidelberg, 2005. – P. 213-261.

44. Stipčević, M. Characterization of a novel avalanche photodiode for single photon detection in VIS-NIR range / M. Stipčević, H. Skenderović, D. Gracin // Optics Express. – 2010. – Vol. 18. – № 16. – P. 17448-17459.

45. Ходжес, М. Модули счета одиночных фотонов на основе лавинных фотодиодов / М. Ходжес, С. Граберг // Фотоника. – 2013. – № 4. – С. 094-099.

46. Becker, W. Advanced time-correlated single photon counting techniques. Vol. 81 / W. Becker. – 2005.

47. Беккер, В. Визуализация времени жизни флуоресценции с помощью многомерного TCSPC-метода: новые возможности в биомедицине / В. Беккер, А. Желзов, В. Чеславский // Фотоника. – 2015. – Т. 53. – № 5. – С. 52.

48. Лазерный флуорометр с наносекундным разрешением для ближнего ИК-диапазона / В. Галиевский, А. Сташевский, В. Киселев [и др.] // Приборы и техника эксперимента. – 2010. – № 4. – С. 109-116.

49. Щеславский, В. Люминесцентная микроскопия на основе многопараметрического время-коррелированного счета фотонов / В. Щеславский, М. Ширманова, А. Ельцов // Успехи биологической химии. – 2019. – Т. 59. – С. 103-138.

50. Селивонин, И. В. Цифровой метод коррелированного по времени счета одиночных фотонов для диагностики барьерного разряда / И. В. Селивонин, С. . Кувардин, И. А. Моралев // Физика плазмы. – 2023. – Т. 49. – № 5. – С. 462-470.

51. 20-ps timing resolution with single-photon avalanche diodes / S. Cova, A. Lacaita, M. Ghioni [et al.] // Review of Scientific Instruments. – 1989. – Vol. 60. – № 6. – P. 1104-1110.

52. Cova, S. Towards picosecond resolution with single-photon avalanche diodes / S. Cova, A. Longoni, A. Andreoni // Review of Scientific Instruments. – 1981. – Vol. 52. – P. 408-412.

53. Lacaita, A. Four-hundred-picosecond single-photon timing with commercially available avalanche photodiodes / A. Lacaita, S. Cova, M. Ghioni // Review of Scientific Instruments. – 1988. – Vol. 59. – № 7. – P. 1115-1121.

54. Prochazka, I. Recent achievements in single photon detectors and their applications / I. Prochazka, K. Hamal, B. Sopko // Journal of Modern Optics. – 2004. – Vol. 51. – № 9-10. – P. 1289-1313.

55. Avalanche photodiodes and quenching circuits for single-photon detection / S. Cova, M. Ghioni, A. Lacaita [et al.] // Applied Optics. – 1996. – Vol. 35. – № 12. – P. 1956-1976.

56. Ekstrom, P. A. Triggered-avalanche detection of optical photons / P. A. Ekstrom // Journal of Applied Physics. – 1981. – Vol. 52. – № 11. – P. 6974-6979.

57. S. Cova. Active-Quenching and Gating Circuits for Single-Photon Avalanche Diodes (SPADs) / S. Cova, A. Longoni, G. Ripamonti // IEEE Transactions on Nuclear Science. – 1982. – Vol. 29. – № 1. – P. 599-601.

58. Hadfield, R. H. Single-photon detectors for optical quantum information applications / R. H. Hadfield // Nature Photonics. – 2009. – Vol. 3. – № 12. – P. 696-705.

59. Advances in InGaAs/InP single-photon detector systems for quantum communication / J. Zhang, M. A. Itzler, H. Zbinden, J.-W. Pan // Light: Science & Applications. – 2015. – Vol. 4. – № 5. – P. e286-e286.

60. Детекторы одиночных фотонов на основе ОЛФД – схемотехнические решения и режимы работы / А. В. Лосев, В. В. Заводиленко, А. А. Козий [и др.] // Микроэлектроника. – 2021. – Т. 50. – № 2. – С. 116-126.

61. Decker, R. W. State-of-the-art in volcano forecasting / R. W. Decker // Bulletin Volcanologique. – 1973. – Т. 37. – № 3. – С. 372-393.

62. Sparks, R. S. J. Forecasting volcanic eruptions / R. S. J. Sparks // Earth and Planetary Science Letters. – 2003. – Т. 210. – № 1. – С. 1-15.

63. Fiorani, L. Measurement of Mount Etna plume by CO2-laser-based lidar / L. Fiorani, F. Colao, A. Palucci // Optics Letters. – 2009. – Vol. 34. – № 6. – P. 800-802.

64. Volcanic ash plume identification using polarization lidar: Augustine eruption, Alaska / K. Sassen, J. Zhu, P. Webley [et al.] // Geophysical Research Letters. – 2007. – Vol. 34. – № 8.

65. First observational evidence for the CO2-driven origin of Stromboli's major explosions / A. Aiuppa, M. Burton, P. Allard [et al.] // Solid Earth. – 2011. – Vol. 2. – № 2. – P. 135-142.

66. Unusually large magmatic CO2 gas emissions prior to a basaltic paroxysm / A. Aiuppa, M. Burton, T. Caltabiano [et al.]. – Text : electronic // Geophysical Research Letters. – 2010. – Vol. 37. – № 17. – URL: https://doi.org/10.1029/2010GL043837 (date accessed: 11.04.2024).

67. The 2007 eruption of Stromboli volcano: insights from real-time measurement of the volcanic gas plume CO2/SO2 ratio / A. Aiuppa, C. Federico, G. Giudice [et al.] // Journal of Volcanology and Geothermal Research. – 2009. – Vol. 182. – № 3-4. – P. 221-230.

68. Moretti, R. A model for the saturation of C-O-H-S fluids in silicate melts / R. Moretti, P. Papale, G. Ottonello // Geological Society, London, Special Publications. – 2003. – Т. 213. – С. 81-101.

69. Queißer, M. Insights into geological processes with CO2 remote sensing – A review of technology and applications / M. Queißer, M. Burton, R. Kazahaya // Earth-Science Reviews. – 2019. – Т. 188. – С. 389-426.

70. Wallace, P. J. Volcanic SO2 emissions and the abundance and distribution of exsolved gas in magma bodies / P. J. Wallace // Journal of Volcanology and Geothermal Research. – 2001. – Т. 108. – № 1. – С. 85-106.

71. Monitoring Quiescent Volcanoes by Diffuse CO2 Degassing: Case Study of Mt. Fuji, Japan / K. Notsu, T. Mori, S. C. D. Vale [et al.] // pure and applied geophysics. – 2006. – Vol. 163. – № 4. – P. 825-835.

72. Volcanic CO2 detection with a DFM/OPA-based lidar / L. Fiorani, S. Santoro, S. Parracino [et al.] // Optics Letters. – 2015. – Vol. 40. – № 6. – P. 1034-1036.

73. New ground-based lidar enables volcanic CO2 flux measurements / A. Aiuppa, L. Fiorani, S. Santoro [et al.] // Scientific Reports. – 2015. – Vol. 5. – № 1. – P. 13614.

74. Queiβer, M. A new frontier in CO2 flux measurements using a highly portable DIAL laser system / M. Queiβer, D. Granieri, M. Burton // Scientific Reports. – 2016. – Vol. 6. – № 1. – P. 33834.

75. Miniature High-Power Nanosecond Laser Diode Transmitters Using the Simplest Possible Avalanche Drivers / S. Vainshtein, V. Zemlyakov, V. Egorkin [et al.] // IEEE Transactions on Power Electronics. – 2019. – Vol. 34. – № 4. – P. 3689-3699.

76. High-Power and Repetion Rate Nanosecond Pulse Generation in “Diode Laser—Thyristor” Stacks / S. O. Slipchenko, A. A. Podoskin, V. S. Golovin [et al.] // IEEE Photonics Technology Letters. – 2021. – Vol. 33. – № 1. – P. 11-14.

77. Tontini, A. Numerical Model of SPAD-Based Direct Time-of-Flight Flash LIDAR CMOS Image Sensors / A. Tontini, L. Gasparini, M. Perenzoni // Sensors. – 2020. – Vol. 20. – № 18.

78. Invited Review Article: Single-photon sources and detectors / M. D. Eisaman, J. Fan, A. Migdall, S. V. Polyakov // Review of Scientific Instruments. – 2011. – Т. 82. – № 7. – С. 071101.

79. Новый режим генерации диодного лазера: 200-пикосекундный фронт наносекундного импульса / С. М. Першин, В. С. Макаров, М. Я. Гришин [и др.] // Квантовая электроника. – 2022. – Т. 52. – № 11. – С. 1050-1053.

80. Pope, R. M. Absorption spectrum (380–700 nm) of pure water. II. Integrating cavity measurements / R. M. Pope, E. S. Fry // Applied Optics. – 1997. – Т. 36. – № 33. – С. 8710-8723.

81. Diode-pumped Nd:YAG master oscillator power amplifier with high pulse energy, excellent beam quality, and frequency-stabilized master oscillator as a basis for a next-generation lidar system / M. Ostermeyer, P. Kappe, R. Menzel, V. Wulfmeyer // Applied Optics. – 2005. – Vol. 44. – № 4. – P. 582-590.

82. Беловолов, М. И. Джиттер и предельная нижняя частота импульсов генерации твердотельного лазера с диодной накачкой при пассивной модуляции

добротности резонатора / М. И. Беловолов, А. Ф. Шаталов // Квантовая электроника. – 2008. – Т. 38. – № 10. – С. 933-934.

83. Underwater navigation: LiDAR sensing in water to 9 m distance through semi-transparent obstacles / S. M. Pershin, M. Ya. Grishin, V. A. Zavozin [et al.] // 2022 International Conference Laser Optics (ICLO) / Citation Key: 9839859. – 2022. – P. 1.

84. Underwater Lidar: Remote Sensing in Strongly Scattering Media / S. M. Pershin, A. F. Bunkin, V. A. Zavozin [et al.] // Physics of Wave Phenomena. – 2023. – Vol. 31. – № 6. – P. 406-411.

85. Eye-safe photon counting LIDAR for magmatic aerosol detection / V. A. Zavozin, M. Y. Grishin, V. N. Lednev [et al.] // Laser Physics. – 2022. – Vol. 32. – № 12. – P. 125601.

86. Volcanic activity monitoring by unique LIDAR based on a diode laser / S. M. Pershin, A. L. Sobisevich, M. Y. Grishin [et al.] // Laser Physics Letters. – 2020. – Vol. 17. – № 11. – P. 115607.

87. Aerosol layers sensing by an eye-safe lidar near the Elbrus summit / S. M. Pershin, M. Y. Grishin, V. A. Zavozin [et al.] // Laser Physics Letters. – 2020. – Vol. 17. – № 2. – P. 026003.

88. Bunkin, A. Lidar measurement of dynamics of spatial characteristics of aerosol in boundary atmospheric layer under urban conditions / A. Bunkin, S. Pershin // Physics of Wave Phenomena. – 2005. – Т. 13. – № 1. – С. 37.

89. Ecological catastrophe in Arctic: An anomalous Gulf Stream heating (to 21 C) and shift (~ 200 km) to Greenland due to ocean pollution by rainbow oil film / I. Bjørnø, M. Grishin, S. Pershin, I. Bjørnø // Proc. 5th Underwater Acoustics Conf. and Exhibition (UACE2019), Hersonissos, Crete, Greece, June 30–July 5. – 2019. – С. 129-138.

90. Laser Remote Sensing of Lake Kinneret by Compact Fluorescence LiDAR / S. M. Pershin, B. G. Katsnelson, M. Ya. Grishin [et al.] // Sensors. – 2022. – Vol. 22. – № 19.

91. Platform effects on optical variability and prediction of underwater visibility / G. Chang, M. S. Twardowski, Y. You [и др.] // Applied Optics. – 2010. – Т. 49. – № 15. – С. 2784-2796.

92. Remote sensing of seawater and drifting ice in Svalbard fjords by compact Raman lidar / A. F. Bunkin, V. K. Klinkov, V. N. Lednev [et al.] // Applied Optics. – 2012. – Vol. 51. – № 22. – P. 5477-5485.

93. Remote sensing of Arctic Fjords by Raman lidar: Heat transfer screening by layer of glacier's relict water / S. M. Pershin, A. F. Bunkin, V. K. Klinkov [et al.] // Physics of Wave Phenomena. – 2012. – Vol. 20. – № 3. – P. 212-222.

94. Zhang, X. Scattering by pure seawater: Effect of salinity / X. Zhang, L. Hu, M.-X. He // Optics Express. – 2009. – Т. 17. – № 7. – С. 5698-5710.

95. Лидарное зондирование эволюции многослойных туманов в наклонном тоннеле Баксанской нейтринной обсерватории / С. М. Першин, А. Н. Федоров, А. В. Тюрин [и др.] // Краткие сообщения по физике ФИАН. – 2019. – Т. 10. – С. 46-54.

96. Лидар дифференциального поглощения для непрерывного мониторинга эмиссии газов в тоннеле вблизи вулкана Эльбрус / Я. Я. Понуровский, С. М. Першин, М. Я. Гришин [и др.] // Инженерная физика. – 2024. – № 1. – С. 3-13.

97. Мельхиор, П. Земные приливы / П. Мельхиор; ред. Н. Н. Парийский; пер. С. Н. Барсенкова, Ю. С. Доброхотова, Б. П. Перцова. – Москва : Издательство «Мир», 1968.

98. Estimation of the Influence of Meteorological Factors on the Aerosol Lidar Signal in Tunnels above the Elbrus Volcano Chamber / A. V. Myasnikov, S. M. Pershin, M. Ya. Grishin [et al.] // Physics of Wave Phenomena. – 2022. – Vol. 30. – № 2. – P. 119-127.

99. Warden, S. Long term air ion monitoring in search of pre-earthquake signals / S. Warden, T. Bleier, K. Kappler // Journal of Atmospheric and Solar-Terrestrial Physics. – 2019. – Vol. 186. – P. 47-60.

100. Observation of an Excess of Positive Air Ions in Underground Cavities / L. Bezrukov, A. Gromtseva, V. Zavarzina [et al.] // Geomagnetism and Aeronomy. – 2022. – Vol. 62. – № 6. – P. 743-755.

Приложение 1. Программа для управления лидаром

Листинг 1. Протокол взаимодействия с лидаром

```
import serial
import serial.tools.list_ports
from lidar.project_log import project_logger
from lidar.protocols import Protocol

class LiDAR_Port(object):
    """Class for LiDAR_Port."""

    def __init__(self, com_port: str, baudrate: int, protocol: Protocol):
        try:
            self.port = serial.Serial(com_port, baudrate, timeout=1.6)
            self.protocol = protocol
            project_logger.debug(f'COM port {com_port} is connected')
        except Exception:
            raise

    def close(self):
        self.port.close()
        project_logger.debug('COM port is closed')
        return

    def read_bytes(self, num_of_bytes: int = 1) -> bytes:
        input_bytes = self.port.read(num_of_bytes)
        project_logger.debug(f'Read {num_of_bytes} bytes')
        return input_bytes

    def set_pulses(
        self,
        pulses: int,
    ) -> bool:
        """Set pulses."""
        project_logger.debug(f'Set pulses {pulses}')
        command = self.protocol.get_pulses_command(pulses)
        project_logger.debug(f'Command for set pulses is {command.hex()}')
        self.port.write(command)
        read_answer = self.port.read(2)
        project_logger.debug(f'Answer is {read_answer.hex()}')
        if command.find(read_answer) != -1 and read_answer != b'':
            project_logger.info(
                f'Pulses are set.\n Find answer in command with index
{command.find(read_answer)}')
            return True
        else:
            project_logger.error(f'Invalid answer {read_answer.hex()}')
            return False

    def read_lidar_data(
            self,

            file,
    ):
        self.port.write(self.protocol.start)
        project_logger.info(
            f'Start measure with command {self.protocol.start.hex()}')
        stop = bytes(0)
        lidar_bytes = bytes(1)
```

```
        stop_index = -1
        buffer = 64
        while not (stop == self.protocol.stop or lidar_bytes == b''):
            lidar_bytes = self.read_bytes(buffer)
            project_logger.debug(f'Read bytes {lidar_bytes.hex()}')
            file.write(lidar_bytes)
            stop_index = lidar_bytes.find(self.protocol.stop)
            project_logger.debug(f'Stop index is {stop_index}')
            if stop_index != -1:
                stop = self.protocol.stop
        if stop_index != -1:
            last_buffer = 12 - buffer + stop_index + 2
            project_logger.debug(f'Number of last bytes {last_buffer}')
            if last_buffer > 0:
                lidar_bytes = self.read_bytes(last_buffer)
                project_logger.debug(f'Last bytes are {lidar_bytes.hex()}')
                file.write(lidar_bytes)

    def read_parameters(self, file):
        project_logger.info(f'Read parameters')
        for key in self.protocol.parameters.keys():
            self.port.write(self.protocol.parameters[key].get('command'))
            project_logger.debug(
                f'Get {key}. Send command
{self.protocol.parameters[key].get("command").hex()}'
            )
            answer = self.port.read(
                self.protocol.parameters[key].get('answer_bytes'))
            project_logger.debug(
                f'Read answer {answer.hex()}'
            )
            file.write(key.encode() + b'\n' + answer + b'\n')


def available_com_ports() -> list:
    result = []
    ports = serial.tools.list_ports.comports()
    for port, _, _ in sorted(ports):
        result.append(port)
    return result
```

Листинг 2. Основная программа

```
import pathlib
import com_port
import PySimpleGUI as sg
import datetime
from lidar.measure import measure
from lidar.project_log import project_logger
from lidar.protocols import protocols
from threading import Thread


lidar_port = None
measurements = False
toggle_btn_off = b'iVBORw0KGgoAAAANSUhEUgAAACgAAAAoCAYAAACM/rhtAAAABmJLR0QA/wD/AP+
=='
toggle_btn_on =
b'iVBORw0KGgoAAAANSUhEUgAAACgAAAAoCAYAAACM/rhtAAAABmJLR0QA/wD/AP+gvaeTAAAD+ +'

if __name__ == "__main__":
```

```
    project_logger.info('Start program')
    sg.theme('DarkAmber')
    layout = [
        [sg.Text('COM Port')],
        [sg.LBox(values=com_port.available_com_ports(),
                 size=(20, 10), key='com_port'),
         sg.LBox(values=[9600, 115200, 460800], size=(20, 10), key='baudrate'),
         sg.LBox(values=list(protocols.keys()), size=(20, 10), key='Protocol')],
        [sg.Text('Off'), sg.Button(image_data=toggle_btn_off, key='Connect',
button_color=(
            sg.theme_background_color(), sg.theme_background_color()),
border_width=0), sg.Text('On')],
        [sg.Text('Папка для сохранения файлов')],
        [sg.Input(key='folder'), sg.FolderBrowse(target='folder')],
        [sg.Text('Количество импульсов\nв измерении'),
         sg.Text('Время и дата\nначала измерений')],
        [sg.LBox(values=[50, 100, 200, 500, 1000, 2000, 5000, 10000, 20000, 50000,
64000, 100000, 200000, 500000, 1000000, 2000000], size=(
            20, 10), key='pulses'),
         sg.Input(default_text=datetime.datetime.now().strftime(
             '%Y-%m-%d %H:%M:%S'), key='date', size=(20, 10)),
         sg.CalendarButton('Выберете дату', close_when_date_chosen=True,
target='date')],
        [sg.Text('Период измерений в минутах'),
         sg.Input(default_text='10', size=(20, 1), key='period')],
        [sg.Text('Количество измерений'),
         sg.Input(default_text='1', size=(20, 1), key='count')],
        [sg.Button('Start'), sg.Button('Start series'),
         sg.Button('Stop'), sg.Button('Exit')]
    ]
    window = sg.Window('Test', layout)
    project_logger.debug(f'Window is generated {window}')

    while True:
        event, values = window.read()
        project_logger.debug(f'Event is {event}')
        project_logger.debug(f'Values is {values}')
        if event == sg.WIN_CLOSED or event == 'Exit':
            if measurements:
                for measurement in measurements:
                    if measurement.is_alive():
                        project_logger.info(
                            f'Measurement {measurement.name} in progress')
                    measurement.join()
            if lidar_port:
                lidar_port.close()
                project_logger.debug('COM port is closed')
            break
        if event == 'Connect':
            if lidar_port:
                lidar_port.close()
                lidar_port = False
                project_logger.debug('COM port is closed')
                window['Connect'].update(
                    image_data=toggle_btn_off
                )
            else:
                try:
                    lidar_port = com_port.LiDAR_Port(
                        values['com_port'][0], (values['baudrate'][0]),
protocol=protocols.get(values['Protocol'][0]))
```

```
                    window['Connect'].update(
                        image_data=toggle_btn_on if lidar_port is not None else
toggle_btn_off)
                    project_logger.debug(f'COM port {lidar_port} is connected')
                except IndexError as index_error:
                    sg.popup_error(
                        'Error', f'Выберете COM порт и скорость\n{index_error}')
                    project_logger.error(
                        f'COM port and baudrate\nNot selected{index_error}')
                    window['Connect'].update(
                        image_data=toggle_btn_off
                    )
                except Exception as e:
                    sg.popup_error('Error',
                                   f'{e}\nCOM {values["com_port"]} baud
{values["baudrate"]}')
                    project_logger.error(
                        f'{e}\nCOM {values["com_port"]} baud
{values["baudrate"]}')
                    window['Connect'].update(image_data=toggle_btn_off)
        if event == 'Start series':
            if lidar_port:
                pass
            else:
                project_logger.error(
                    'COM port not connected.Can not start measurements')
                sg.popup_error('Error', 'COM port not connected')
        if event == 'Start':
            if lidar_port:
                try:
                    counts = int(values['count'])
                    pulses = int(values['pulses'][0])
                    folder = pathlib.Path(values['folder'])
                except Exception as e:
                    sg.popup_error('Error', f'{e}')
                    project_logger.error(f'{e}')
                    continue
                project_logger.debug('Start measurements')
                measurements = [Thread(target=measure, name='Measure #' + str(
                    i), args=(lidar_port, folder, pulses)) for i in range(counts)]
                for measurement in measurements:
                    measurement.start()

            else:
                project_logger.error(
                    'COM port not connected.Can not start measurements')
                sg.popup_error('Error', 'COM port not connected')
        if event == 'Stop':
            pass
    project_logger.debug('Window is closed')
    window.close()
```

Листинг 3. Интерфейс командной строки CLI (англ. command line interface)

```
from lidar.com_port import LiDAR_Port
import schedule
from os import PathLike
from lidar.protocols import protocols
from lidar.measure import measure
from lidar.project_log import project_logger
```

```
import click

@click.command()
@click.option('--com_port', '-c', type=str, default='COM3', help='COM port')
@click.option('--baudrate', '-b', type=int, default=115200, help='Baudrate')
@click.option('--protocol', '-p', type=str, default='delft', help='Protocol')
@click.option('--folder', '-f', type=click.Path(exists=True, file_okay=False,
dir_okay=True), default='.', help='Folder')
@click.option('--pulses', '-pulses', type=int, default=5000, help='Pulses int')
@click.option('--period', '-period', type=int, default=10, help='Period. Interval
in minutes.')
def LiDAR_cli(
    com_port: str,
    baudrate: int,
    protocol: str,
    folder: PathLike,
    pulses: int = 50,
    period: int = 10
) -> None:
    project_logger.info(f'Input COM port {com_port}')
    project_logger.info(f'Input Baudrate {baudrate}')
    project_logger.info(f'Input protocol {protocol}')
    project_logger.info(f'Input Folder {folder}')
    project_logger.info(f'Input Pulses {pulses}')
    project_logger.info(f'Input Period {period}')
    lidar_protocol = protocols.get(protocol)
    project_logger.info(f'Protocol for LiDAR_Port {lidar_protocol}')
    lidar = LiDAR_Port(com_port, baudrate, lidar_protocol)
    list_of_times = [':'+str(minutes).zfill(2)
                     for minutes in range(0, 60, period)]
    for time in list_of_times:
        project_logger.info(f'Time {time}')
        schedule.every().hour.at(time).do(measure, lidar, folder, pulses)
    while True:
        try:
            schedule.run_pending()
        except Exception as error:
            project_logger.error(error)
            lidar.close()
            project_logger.debug('COM port is closed')
            break
```

Листинг 4. Измерение гистограммы временных задержек между стартом генерации стартового импульса и регистрацией фотона SPAD

```
from datetime import datetime
import lidar.com_port as com_port
import os
import pathlib
from lidar.project_log import project_logger
from threading import BoundedSemaphore

connections = BoundedSemaphore()

def measure(
        port: com_port.LiDAR_Port,
        folder: os.PathLike,
        pulses: int = 50
) -> None:
```

```
    """Measure."""
    connections.acquire()
    project_logger.info(f'Start measurement')
    filename = datetime.now().strftime("%Y-%m-%d_%H-%M-%S") + '.bin'
    path = pathlib.Path(folder, filename)
    project_logger.info(f'File in {path}')
    try:
        file = open(path, 'wb')
        project_logger.info(f'File {path} is opened')
        file.write(
            b'Datetime ' + datetime.now().strftime("%Y-%m-%d %H:%M:%S").encode() +
b'\n')
        file.write(b'LiDAR ' + port.protocol.name.encode() + b'\n')
        file.write(b'Pulses ' + str(pulses).encode() + b'\n')
        file.write(b'Data\n')
    except Exception as error:
        project_logger.error(error)
        raise
    if port.set_pulses(pulses):
        port.read_lidar_data(file)
        file.write(b'\nEnd of data;\n')
        file.write(b'Parameters\n')
        project_logger.info(f'Data collected')
        port.read_parameters(file)
        file.write(b'\nEnd of parameters;')
        project_logger.info(f'Parameters collected')
        project_logger.info(f'Measure is done')
        file.close()
        project_logger.info(f'File {path} is closed')
    else:
        project_logger.error(f'Measure can\'t set pulses. Abort.')
        file.close()
        project_logger.info(f'File {path} is closed')
    connections.release()
    return
```

Листинг 5. Проектный логгер

```
from loguru import logger
import sys
project_logger = logger
logger.configure(handlers=[{"sink": sys.stderr, "level": "INFO"}])
project_logger.add(
    "file{time}.log", rotation="1 week", retention=5, format="{time} {level}
{module} {name} {message}", level="DEBUG", )
```

Листинг 6. Команды для взаимодействия с микроконтроллером лидара

```
from project_log import project_logger
import copy
class Protocol(object):
    def __init__(self, name: str):
        self.name = name
        self.pulses_commands = {}
        self.start = bytes()
        self.stop = bytes()
        self.SPAD_voltage = bytes()
        self.parameters = {}
```

```
        self.read_data_protocol = {}

    def get_pulses_command(self, pulses: int) -> bytes:
        if self.pulses_commands != {}:
            try:
                return self.pulses_commands.get(pulses)
            except Exception:
                raise
        else:
            raise NotImplemented

    def set_spad_voltage(self, voltage: int):
        if self.set_spad_voltage != bytes:
            try:
                return self.SPAD_voltage + bytes(voltage)
            except Exception:
                raise
        else:
            raise NotImplemented

    def read_parameters(self):
        if self.parameters != {}:
            try:
                return self.parameters
            except Exception:
                raise
        else:
            raise NotImplemented

    def read_data_from_file(self, lidar_file):
        result = copy.deepcopy(self.read_data_protocol)
        if lidar_file.readline() == b'Data\n':
            read_bytes = lidar_file.read(2)
            project_logger.debug(f'Read bytes: {read_bytes.hex()}')
            while read_bytes != bytes.fromhex('FFFF'):
                project_logger.debug(f'Read bytes: {read_bytes.hex()}')
                for key in result.keys():
                    project_logger.debug(f'Key: {key}')
                    if read_bytes == result[key].get('mask'):
                        read_bytes = lidar_file.read(
                            result[key].get('number_of_bytes'))
                        project_logger.debug(
                            f'Data bytes: {read_bytes.hex()},
{int.from_bytes(read_bytes, byteorder="big")}')
                        result[key].get('delays').append(
                            int.from_bytes(read_bytes, byteorder='big'))
                        break
                    elif result[key].get('mask') is None:
                        project_logger.debug(
                            f'Data bytes: {read_bytes.hex()},
{int.from_bytes(read_bytes, byteorder="big")}')
                        result[key].get('delays').append(
                            int.from_bytes(read_bytes, byteorder='big'))
                        break
                read_bytes = lidar_file.read(2)

        else:
            raise NotImplemented
        return result
hokkaido = Protocol("hokkaido")
hokkaido.pulses_commands = {
```

```
    50: bytes.fromhex('F5 F5 70'),
    100: bytes.fromhex('F5 F5 71'),
    200: bytes.fromhex('F5 F5 72'),
    500: bytes.fromhex('F5 F5 73'),
    1000: bytes.fromhex('F5 F5 74'),
    2000: bytes.fromhex('F5 F5 75'),
    5000: bytes.fromhex('F5 F5 76'),
    10000: bytes.fromhex('F5 F5 77'),
    20000: bytes.fromhex('F5 F5 78'),
    50000: bytes.fromhex('F5 F5 79'),
    65536: bytes.fromhex('F5 F5 7A'),
    100000: bytes.fromhex('F5 F5 7B'),
    200000: bytes.fromhex('F5 F5 7C'),
    500000: bytes.fromhex('F5 F5 7D'),
    1000000: bytes.fromhex('F5 F5 7E'),
    2000000: bytes.fromhex('F5 F5 7F')
}
hokkaido.start = bytes.fromhex('F5 F5 51')
hokkaido.stop = bytes.fromhex('FF FF')
hokkaido.parameters = {
    'number_of_stops': {'command': bytes.fromhex('F5 F5 52'), 'answer_bytes': 6}
}
hokkaido.SPAD_voltage = bytes.fromhex('FA FA')
hokkaido.read_data_protocol = {
    'channel1': {'mask': None, 'number_of_bytes': 2, 'delays': list()}
}

delft = Protocol("delft")
delft.pulses_commands = {
    50: bytes.fromhex('F5 F5 70'),
    100: bytes.fromhex('F5 F5 71'),
    200: bytes.fromhex('F5 F5 72'),
    500: bytes.fromhex('F5 F5 73'),
    1000: bytes.fromhex('F5 F5 74'),
    2000: bytes.fromhex('F5 F5 75'),
    5000: bytes.fromhex('F5 F5 76'),
    10000: bytes.fromhex('F5 F5 77'),
    20000: bytes.fromhex('F5 F5 78'),
    50000: bytes.fromhex('F5 F5 79'),
    64000: bytes.fromhex('F5 F5 7A'),
    100000: bytes.fromhex('F5 F5 7B'),
    200000: bytes.fromhex('F5 F5 7C'),
    500000: bytes.fromhex('F5 F5 7D'),
    1000000: bytes.fromhex('F5 F5 7E'),
    2000000: bytes.fromhex('F5 F5 7F')
}
delft.start = bytes.fromhex('F5 F5 51')
delft.stop = bytes.fromhex('FF FF')
delft.parameters = {
    'energy': {'command': bytes.fromhex('F5 F5 54'), 'answer_bytes': 5},
    'temperature': {'command': bytes.fromhex('F5 F5 55'), 'answer_bytes': 4},
    'Ch1': {'command': bytes.fromhex('F5 F5 01'), 'answer_bytes': 5},
    'Ch2': {'command': bytes.fromhex('F5 F5 02'), 'answer_bytes': 5}
}
delft.SPAD_voltage = bytes.fromhex('FA FA')
delft.read_data_protocol = {
    'channel1': {'mask': bytes.fromhex('EBEB'), 'number_of_bytes': 2, 'delays':
list()},
    'channel2': {'mask': bytes.fromhex('EDED'), 'number_of_bytes': 2, 'delays':
list()},
```

```
    'noise1': {'mask': bytes.fromhex('EAEA'), 'number_of_bytes': 2, 'delays':
list()},
    'noise2': {'mask': bytes.fromhex('ECEC'), 'number_of_bytes': 2, 'delays':
list()}
}
protocols = {
    'hokkaido': hokkaido,
    'delft': delft
}
```

Листинг 7. Чтение и обработка результатов измерения из файла

```
import asyncio
import os
from datetime import datetime
from pathlib import Path
import re
import sys
from project_log import project_logger
from protocols import protocols
import pandas as pd
import tqdm.asyncio
from tqdm import tqdm

def read_header(lidar_file):
    """ Read the header of the lidar file """
    datetime_str = lidar_file.readline().decode().strip()
    hist_datetime = datetime.strptime(
        datetime_str, 'Datetime %Y-%m-%d %H:%M:%S')
    project_logger.info(f'Histogram datetime: {hist_datetime}')
    project_logger.debug(
        f'Histogram datetime: {hist_datetime} from {datetime_str}')

    lidar_name_str = lidar_file.readline().decode().strip()
    lidar_name = re.findall(r'(?<=LiDAR ).*', lidar_name_str)[0]
    project_logger.info(f'Lidar name: {lidar_name}')
    project_logger.debug(f'Lidar name: {lidar_name} from {lidar_name_str}')

    pulses_str = lidar_file.readline().decode().strip()
    project_logger.debug(f'Pulses str: {pulses_str}')
    pulses = re.findall(r'(?<=Pulses )\d+', pulses_str)[0]
    pulses = int(pulses)
    project_logger.info(f'Pulses: {pulses}')

    return hist_datetime, lidar_name, pulses

def signals(
        result_file,
        channel1: pd.Series,
        channel2: pd.Series,
        hist_datetime: datetime
):
    aerosol_bounds = [70, 179]
    target_bounds = [180, 290]
    df_signals = pd.DataFrame(
        {
            0: hist_datetime,
            1: channel1.loc[aerosol_bounds[0]:aerosol_bounds[1]].sum(),
            2: channel1.loc[target_bounds[0]:target_bounds[1]].sum(),
            3: channel2.loc[aerosol_bounds[0]:aerosol_bounds[1]].sum(),
            4: channel2.loc[target_bounds[0]:target_bounds[1]].sum()
        },
```

```
        index=[0]
    )
    df_signals.to_csv(
        result_file,
        header=False,
        index=False,
        mode='a',
        sep=';',
        date_format='%Y-%m-%d %H:%M:%S'
    )
async def read_file(
    result_file,
    file_dir,
    file_name
):
    project_logger.info(f'Reading file {file_name}')
    file = open(os.path.join(file_dir, file_name), 'rb')
    histogram_datetime, lidar_name, pulses = read_header(file)
    lidar_device = protocols[lidar_name]
    data = lidar_device.read_data_from_file(file)
    file.close()
    signals(
        result_file,
        pd.Series(data['channel1'].get('delays')
                  ).value_counts().sort_index(),
        pd.Series(data['channel2'].get('delays')
                  ).value_counts().sort_index(),
        histogram_datetime
    )
async def signals_from_dir(
    result_file,
        dir_f
):
    files = os.listdir(dir_f)
    for f in tqdm(files):
        await read_file(result_file, dir_f, f)
    return
```

Приложение 2. Программа для автоматического мониторинга вариации аэрозолей, концентрации газов и метеопараметров в штольне Баксанской Нейтринной Обсерватории

Листинг 1. Проектный логгер

```
from loguru import logger
log = logger
log.add("./logs/bno_monitoring.log", level="INFO", rotation="2 days")
```

Листинг 2. Графическое представление данных

```
import asyncio
from plotly import express
from plotly.graph_objs import Figure
from plotly.subplots import make_subplots
import plotly.graph_objects as go
from datetime import datetime
from bno_monitoring.sql_connect import engine
from bno_monitoring.read_sql_data import read_all_data

def make_traces_order_by_name(fig, tup: list, data: dict):
    colors = express.colors.qualitative.Dark24 + express.colors.qualitative.Dark24
    for key in enumerate(tup):
        sensor = key[1].split(",")
        fig.add_trace(
            go.Scatter(
                x=data[sensor[0]]["data"].index,
                y=data[sensor[0]]["data"]["value"],
                name=key[1],
                mode="lines",
                line_color=colors[key[0]],
            ),
            row=key[0] + 1,
            col=1,
        )
        fig.add_annotation(
            text=f"<b>{key[1]}</b>",
            xref="paper",
            yref="paper",
            x=0.2,
            y=1 - ((key[0]) / len(tup) + 0.02),
            showarrow=False,
            font={"size": 16, "color": colors[key[0]]},
            bgcolor="white",
            opacity=0.9,
        )
    return fig

def all_data_fig(data: dict, names: list) -> Figure:
    fig = make_subplots(rows=len(names), cols=1, shared_xaxes=True,
vertical_spacing=0)
    fig = make_traces_order_by_name(fig, names, data)
    fig.update_xaxes(showgrid=True, gridcolor="darkgray")
    fig.update_yaxes(type="log")
    fig.update_traces(xaxis="x" + str(int(len(names))))
    fig.update_layout(
```

```
        font={"family": "Arial", "size": 14},
        height=1000,
        width=940,
        template="simple_white",
        yaxis={"autorange": True, "fixedrange": False},
        xaxis={"type": "date"},
        showlegend=False,
    )
    return fig
```

Листинг 3. Добавление данных в SQL базу данных

```
import asyncio
from datetime import datetime, timedelta
from pathlib import Path
from bno_monitoring.meteostaion_class import (
    add_new_data_to_db,
    latest_measurement,
    select_meteostation,
)
from bno_monitoring.ldp_class import add_new_data_in_dir_to_db,
latest_ldp_measurement
from bno_monitoring.lidar_class import (
    latest_histogram_in_db,
    lidar_monitoring,
    select_lidar,
)
from bno_monitoring.bno_logger import log
from bno_monitoring.bot import send_message, token

async def monitoring(engine, meteo_file: Path, ldp_dir: Path, lidar_dir: Path):
    await add_new_data_to_db(
        engine, meteo_file, meteostation="Ушаков", meteostation_version="2.0"
    )
    await add_new_data_in_dir_to_db(engine, ldp_dir)
    await lidar_monitoring(
        engine, "BNO*.lvm", lidar="Elbrus", lidar_version="2.0", dir=lidar_dir
    )

async def alert(engine):
    latest_reg = {}
    latest_reg["Метеостанция"] = await latest_measurement(
        engine,
        await select_meteostation(
            engine, meteostation="Ушаков", meteostation_version="2.0"
        ),
    )

    log.info(f"Последние данные в БД. {latest_reg}")
    for key in latest_reg.keys():
        if datetime.now() - latest_reg[key] > timedelta(minutes=30):
            mes = f"{key}: нет данных в течение получаса, последняя регистрация
{latest_reg[key]}"
            log.error(mes)
```

Листинг 4. Сбор данных с лидара дифференциального поглощения

```
import asyncio
from datetime import datetime
from pathlib import Path
```

```
from pandas import pandas
from sqlalchemy import Insert, desc, select
from sqlalchemy.orm import Session

from bno_monitoring.sql_connect import LDP_data
from bno_monitoring.bno_logger import log

async def read_data_file(file: Path) -> pandas.DataFrame:
    log.debug(f"Чтение ldp из файла {file}")
    try:
        ldp_data = pandas.read_csv(
            file,
            sep=r"\s+",
            header=None,
            names=(
                "Date",
                "Time",
                "CH4",
                "12CO2",
                "13CO2",
                "H2S",
                "NH3",
                "H2O",
                "Pressure",
                "Temperature",
                "13/12",
            ),
            decimal=",",
        )
    except:
        raise
    return await set_datetime_index(ldp_data.fillna(0))

async def set_datetime_index(data: pandas.DataFrame) -> pandas.DataFrame:
    """
    Расчет индекса datetime для данных где есть колнки Date и Time
    """
    data["Datetime"] = [
        datetime.strptime(d + " " + t, "%d.%m.%Y %H:%M:%S")
        for d, t in zip(data["Date"].values, data["Time"].values)
    ]
    data = data.drop(columns=["Date", "Time"])
    data = data.set_index("Datetime")
    log.debug("Рассчет индекса ldp DataFrame")
    return data

async def add_data_to_db(engine, data: pandas.DataFrame) -> None:
    with Session(engine) as session:
        for row in data.itertuples():
            for value, name, units in zip(
                row[1:], data.columns, ["ppm"] * 6 + ["mmHg", "°C", None]
            ):
                log.debug(
                    f"Добавление в LDP_data с игнором {row[0]}, {name}, {value},
{units}"
                )
                session.execute(
                    Insert(LDP_data)
                    .values(
```

```
                        measurement_date=row[0], sensor=name, value=value,
units=units
                    )
                    .prefix_with("IGNORE")
                )
                session.commit()

async def new_measurements(engine, measurements: pandas.DataFrame) ->
pandas.DataFrame:
    return pandas.DataFrame(measurements[await latest_ldp_measurement(engine) :])


async def latest_ldp_measurement(engine) -> datetime:
    with Session(engine) as session:
        result = session.scalar(
            select(LDP_data.measurement_date)
            .order_by(desc(LDP_data.measurement_date))
            .limit(1)
        )
    log.debug(f"Последнее измерение LDP в бд {result}")
    if result is None:
        raise ValueError("No LDP data in db")
    return result

async def measurements_in_dir(dir: Path) -> pandas.Series:
    result = pandas.Series()
    files = list(dir.glob(r"[0-9][0-9].[0-9][0-9].[0-9][0-9][0-9][0-9].txt"))
    if not files:
        raise FileNotFoundError(f"No files with pattern %d.%m.%Y.txt in dir
{dir.name}")
    for file in files:
        result[datetime.strptime(file.name, "%d.%m.%Y.txt")] = file
    log.debug(f"Найденные файлы в папке {result.sort_index()}")
    return result.sort_index()

async def new_files(engine, files: pandas.Series) -> pandas.Series:
    last_date = await latest_ldp_measurement(engine)
    log.info(f"Последнее измерение LDP в бд {last_date}")
    last_date = datetime(year=last_date.year, month=last_date.month,
day=last_date.day)
    new_files = pandas.Series(files[last_date:])
    log.debug(f"Новые файлы в папке {new_files}")
    return new_files

async def add_new_data_in_dir_to_db(engine, dir: Path):
    files = await measurements_in_dir(dir)
    files = await new_files(engine, files)
    await asyncio.gather(
        *[
            add_data_to_db(
                engine, await new_measurements(engine, await read_data_file(f))
            )
            for f in files
        ]
    )
```

Листинг 5. Сбор данных с аэрозольного лидара

```
from datetime import datetime, timedelta
import fnmatch
from pathlib import Path
```

```
import re
import typing
from pandas import DataFrame, pandas
import os
from sqlalchemy import select
from sqlalchemy.orm import Session
from sql_connect import Data, Histogram, Lidar, Lidars_Signals, check_insert
import asyncio
from bno_monitoring.bno_logger import log

async def lidar_files_in_dir(path: Path, pattern: str) -> list:
    """
    Search lidar files in folder with pattern and return list of files
    """
    files = fnmatch.filter(os.listdir(path), pattern)
    files = [os.path.join(path, i) for i in files]
    log.debug(f"Файлы лидара в папке {files}")
    return files

async def file_datetime(
    file: os.PathLike,
    pattern: str,
) -> dict:
    """
    Return datetime of lidar file
    """
    try:
        date_pattern = pattern.replace("*", "_%y-%m-%d_%H%M")
        log.debug(f"Шаблон даты и времени {date_pattern} для файла {file}")
    except:
        print(f"No file with pattern {pattern}")
        raise
    file_datetime = datetime.strptime(os.path.basename(file), date_pattern)
    return dict(file=file, date=file_datetime)

async def search_file_with_histogramm(
    files: list, pattern: str, histogram_date: datetime, period: int, wl: int =
4096
) -> tuple:
    """
    Search file with histogramm in lidar files by date.
    """
    log.debug(f"Дата {histogram_date} шаблон{pattern}, период {period}, wl {wl}")
    files_dates = await asyncio.gather(*[file_datetime(i, pattern) for i in
files])
    file_with_hist = files_dates[0]
    minimum = timedelta(days=2)
    for i in files_dates:
        if i["date"] <= histogram_date:
            if abs(i["date"] - histogram_date) < minimum:
                file_with_hist = i
                minimum = abs(file_with_hist["date"] - histogram_date)
                print(file_with_hist, minimum)
    if abs(file_with_hist["date"] - histogram_date) > timedelta(days=1):
        raise ValueError(f"No file with {histogram_date}")
    data = pandas.read_csv(
        file_with_hist["file"], sep="\t", header=21, usecols=["Comment"]
    )
    data = data.dropna()
    for value in data["Comment"].values:
        hist_date = re.search(r"\d\d\.\d\d\.\d{4} \d+:\d\d:\d\d", value)
```

```
        if hist_date is not None:
            log.debug("В коментариях найдены даты")
            hist_date = datetime.strptime(hist_date[0], "%d.%m.%Y %H:%M:%S")
            if hist_date is not None:
                data.loc[data["Comment"] == value, "Comment"] = hist_date
                hist_date = None
            else:
                raise ValueError(f"Wrong date pattern in file {file_with_hist}")
        else:
            log.warning("Не удалось найти комментарии с датой")
            data = None
            break
    if data is not None:
        hist_date = min(
            [i for i in data["Comment"].values if i <= histogram_date],
            key=lambda x: abs(x - histogram_date),
        )
        log.debug(
            f"Наден ближайший коментарий с датой {hist_date} для {histogram_date}"
        )
        if abs(hist_date - histogram_date) > timedelta(days=1):
            raise ValueError(
                f"No histogram with {histogram_date} in file {file_with_hist}"
            )
        histogram_position = data[data["Comment"] == hist_date].index[0]
    else:
        log.debug(f"Рассчет позиции по периоду {period}")
        histogram_position = (histogram_date - file_with_hist["date"]) //
timedelta(
            minutes=period
        )
        if histogram_position < 0:
            histogram_position = histogram_position + (
                timedelta(days=1) // timedelta(minutes=period)
            )
        histogram_position = histogram_position * wl
    log.info(
        f"Гистограмма с датой {histogram_date} в файле {file_with_hist} с позицией
{histogram_position}"
    )
    return file_with_hist["file"], histogram_position

async def histograms_counts_and_decimal(file: Path, wl: int = 4096) -> tuple:
    """
    Define decimal in histogramm in lidar file
    """
    with open(file, "r") as f:
        lines = f.readlines()
    if lines[-1].find(",") != -1:
        decimal = ","
    else:
        decimal = "."
    histogramm_counts = (len(lines) - 22) // wl
    log.debug(
        f"Кол-во гистограмм {histogramm_counts}, разделитель {decimal}, файл
{file}"
    )
    return histogramm_counts, decimal

async def parse_date_and_pulses(
    lidar_dataframe: DataFrame,
```

```
    histogramm_position: int,
    file_date: datetime,
    period: int,
) -> tuple:
    """
    Parse date and pulses from lidar dataframe
    """
    column_name = lidar_dataframe.iloc[0,
lidar_dataframe.columns.get_loc("Comment")]
    date = re.search(r"\d\d\.\d\d\.\d{4} \d+:\d\d:\d\d", column_name)
    if date is not None:
        date = datetime.strptime(date[0], "%d.%m.%Y %H:%M:%S")
    else:
        date = file_date + timedelta(minutes=period * histogramm_position)
    pulses = re.search(r"\d+$", column_name)
    if pulses is not None:
        pulses = int(pulses[0])
    else:
        pulses = 0
    log.debug(f"Дата {date} и кол-во импульсов {pulses} для гистограммы.")
    return date, pulses

async def histogram_dates_in_file(
    file: Path, filename_pattern: str, period: int
) -> dict:
    """
    Список всех дат измерений гистограм лидаром
    """
    file_date = await file_datetime(file, filename_pattern)
    result = dict()
    list_of_dates = pandas.read_csv(file, header=21, sep="\t",
usecols=["Comment"])
    list_of_dates = list_of_dates.dropna()
    if len(list_of_dates) > 0:
        file_dates = []
        for value in list_of_dates["Comment"].values:
            value = re.search(r"\d\d.\d\d.\d{4} \d+:\d\d:\d\d", value)
            if value is not None:
                file_dates.append(datetime.strptime(value[0], "%d.%m.%Y
%H:%M:%S"))
    else:
        histogramm_counts, _ = await histograms_counts_and_decimal(file)
        file_dates = [
            file_date["date"] + timedelta(minutes=i * period)
            for i in range(histogramm_counts)
        ]
    result[file] = file_dates
    log.debug(f"В файле {file} найдены даты {result[file]}")
    return result

async def read_histogram_from_file(
    file: Path,
    histogramm_position: int,
    decimal: str,
    wl: int = 4096,
):
    """
    Read histogramm from lidar file
    """
    try:
        histogram = await asyncio.to_thread(
```

```
            pandas.read_csv,
            file,
            header=21,
            sep="\t",
            skiprows=list(range(22, histogramm_position + 22)),
            nrows=wl,
            decimal=decimal,
        )
        log.info(
            f"Прочитана гистограмма с позицией {histogramm_position} из файла
{file}"
        )
    except IndexError:
        print(f"No spectr in {file} with histogramm position
{histogramm_position}")
        raise
    histogram = histogram.drop(columns=["X_Value", "Noise"])
    if "Range [Cm]" in histogram.columns:
        histogram.rename(columns={"Range [Cm]": "Distance [Cm]"}, inplace=True)
    histogram = histogram.set_index("Distance [Cm]")
    file_date = await file_datetime(file, "BNO*.lvm")
    date, pulses = await parse_date_and_pulses(
        histogram, histogramm_position, file_date["date"], 10
    )
    histogram = histogram.drop(columns=["Comment"])
    histogram.index.name = "distance"
    histogram.columns = ["signals"]
    histogram = histogram[histogram["signals"] > 0]
    histogram["units"] = "cm"
    log.info(f"Кол-во импульсов {pulses}")
    return histogram, date, pulses

async def measurement(
    engine,
    file,
    lidar: typing.Union[str, int],
    lidar_version: typing.Optional[str] = None,
    histogram_position: int = 1,
    wl: int = 4096,
):
    _, decimal = await histograms_counts_and_decimal(file)
    histogram, date, pulses = await read_histogram_from_file(
        file, histogram_position, decimal, wl=wl
    )
    sql_lidar = await select_lidar(engine, lidar, lidar_version)
    sql_histogram = await select_histogram(engine, sql_lidar, date, pulses)
    histogram["histogram_id"] = sql_histogram.id
    log.info(
        f"Измерение {sql_histogram.id} {sql_histogram.measurement_date} лидар
{sql_lidar.name} {sql_lidar.version} подготовлено"
    )
    return dict(lidar=sql_lidar, histogram=sql_histogram, data=histogram)

async def select_histogram(
    engine,
    lidar: Lidar,
    date: datetime,
    pulses: int,
) -> Histogram:
    with Session(engine) as session:
        sql_histogram = session.scalar(
```

```
            select(Histogram).where(
                Histogram.lidar_id == lidar.id,
                Histogram.measurement_date == date,
                Histogram.pulses == pulses,
            )
        )
        if sql_histogram is None:
            sql_histogram = Histogram(
                lidar_id=lidar.id, measurement_date=date, pulses=pulses
            )
            session.add(sql_histogram)
            session.commit()
            session.refresh(sql_histogram)
    log.debug(
        f"Гистограмма в БД {sql_histogram.id} {sql_histogram.measurement_date}
{sql_histogram.lidar_id} {sql_histogram.pulses}"
    )
    return sql_histogram

async def select_lidar(
    engine, lidar: typing.Union[str, int], lidar_version: typing.Optional[str] =
None
) -> Lidar:
    with Session(engine) as session:
        if isinstance(lidar, int):
            sql_lidar = session.scalar(select(Lidar).where(Lidar.id == lidar))
        else:
            if lidar_version is not None:
                sql_lidar = session.scalar(
                    select(Lidar).where(
                        Lidar.name == lidar, Lidar.version == lidar_version
                    )
                )
            else:
                raise ValueError("lidar_version is required if lidar is str")
        if sql_lidar is None:
            if isinstance(lidar, str):
                if lidar_version is None:
                    raise ValueError("lidar_version is required if lidar is str")
                sql_lidar = Lidar(name=lidar, version=lidar_version)
            else:
                raise ValueError(f"No lidar in db with id {lidar}")
            session.add(sql_lidar)
            session.commit()
            session.refresh(sql_lidar)
    log.debug(
        f"Лидар из БД {sql_lidar.id} {sql_lidar.name} {sql_lidar.name}
{sql_lidar.version}"
    )
    return sql_lidar

async def latest_histogram_in_db(engine, lidar: Lidar) -> Histogram:
    with Session(engine) as session:
        sql_histogram = session.scalar(
            select(Histogram)
            .where(Histogram.lidar_id == lidar.id)
            .order_by(Histogram.measurement_date.desc())
            .limit(1)
        )
        if sql_histogram is None:
            raise ValueError(f"No histogram in db for lidar {lidar}")
```

```
    log.info(
        f"Последняя гистограмма в БД {sql_histogram.id}
{sql_histogram.measurement_date}"
    )
    return sql_histogram

async def files_not_in_db(
    files: typing.List[Path], latest_date: datetime, pattern: str
) -> typing.List[Path]:
    files_with_dates = await asyncio.gather(
        *[file_datetime(file, pattern) for file in files]
    )
    return [
        f["file"]
        for f in files_with_dates
        if f["date"] >= latest_date - timedelta(days=1)
    ]


async def histogram_to_sql(engine, histogram: dict):
    histogram["data"].to_sql("Data", engine, if_exists="append",
method=check_insert)
    log.info(
        f'Загружены данные для гистограммы. {histogram["histogram"].id}
{histogram["histogram"].measurement_date}'
    )
    return histogram["histogram"].id

async def add_data_from_file_to_db(engine, file: Path, lidar: Lidar, wl: int =
4096):
    log.info(
        f"Добавление гистограмм из файла {file} в БД. Лидар {lidar.name}
{lidar.version}"
    )
    histogram_counts, _ = await histograms_counts_and_decimal(file)
    hist_measurements = await asyncio.gather(
        *[
            measurement(engine, file, lidar.id, histogram_position=i, wl=wl)
            for i in range(0, histogram_counts * wl, wl)
        ]
    )
    return await asyncio.gather(
        *[histogram_to_sql(engine, hist) for hist in hist_measurements]
    )


async def insert_signals_in_db(engine, bounds: typing.Sequence, histogram_id:
int):
    df = pandas.read_sql(
        str(
            select(Data.distance, Data.signals)
            .where(Data.histogram_id == histogram_id)
            .compile(engine, compile_kwargs={"literal_binds": True})
        ),
        engine,
        index_col="distance",
    )
    for signal in bounds:
        value = df["signals"].loc[signal.start : signal.stop].sum()
        signal = Lidars_Signals(
            histogram_id=histogram_id,
```

```
            name=signal.name,
            start=signal.start,
            stop=signal.stop,
            value=value,
        )
        with Session(engine) as session:
            check = session.scalar(
                select(Lidars_Signals).where(
                    Lidars_Signals.histogram_id == signal.histogram_id,
                    Lidars_Signals.name == signal.name,
                )
            )
            if not check:
                log.info(
                    f"Добавление сигнала лидара в БД  {signal.histogram_id}
{signal.name} {signal.start}:{signal.stop} = {signal.value}"
                )
                session.add(signal)
                session.commit()

async def lidar_monitoring(
    engine,
    pattern: str,
    lidar: typing.Union[str, int],
    lidar_version: typing.Optional[str],
    dir: Path,
):
    files = await lidar_files_in_dir(
        dir,
        pattern,
    )
    sql_lidar = await select_lidar(engine, lidar, lidar_version)
    last_hist = await latest_histogram_in_db(engine, sql_lidar)
    files = await files_not_in_db(files, last_hist.measurement_date, pattern)
    for file in files:
        await add_data_from_file_to_db(engine, file, sql_lidar)
    with Session(engine) as session:
        last_signals_hist = session.execute(
            select(Lidars_Signals.histogram_id, Histogram.measurement_date)
            .join(Histogram, Histogram.id == Lidars_Signals.histogram_id)
            .order_by(Histogram.measurement_date.desc())
            .limit(1)
        ).first()
        log.info(f"Последний зарегистрированный сигнал {last_signals_hist}")
        if not last_signals_hist:
            raise ValueError("No signals in db")
        last_signals = session.scalars(
            select(Lidars_Signals)
            .join(Histogram, Histogram.id == Lidars_Signals.histogram_id)
            .where(Histogram.id == last_signals_hist[0])
            .distinct()
        ).all()
        ids = session.scalars(
            select(Histogram.id).where(
                Histogram.measurement_date > last_signals_hist[1]
            )
        ).all()
    await asyncio.gather(
        *[insert_signals_in_db(engine, last_signals, id) for id in ids]
    )
```

Листинг 6. Основная программа

```
import asyncio
```

```
from datetime import date, datetime, timedelta
import locale
from pathlib import Path
import dash
from dash import State, dcc
from dash import html
from dash.dependencies import Input, Output
from bno_monitoring.read_sql_data import read_all_data, read_all_sensors,
read_dvo_data
from bno_monitoring.sql_connect import engine
import bno_monitoring.fig_ploting as plot

locale.setlocale(locale.LC_ALL, ("ru_RU", "UTF-8"))
app = dash.Dash(__name__)
server = app.server
app.title = "LiDAR monitoring in BNO"

"""
Read data
"""
spectr_path = Path(
    "/ _Эльбрус_40_пикет/"
)
left_meteo_U = Path(
    "/ метео_В2_40пикет/2023_09_21.txt"
)
gases_path = Path("/_Gases/")
files_pattern = "BNO*.lvm"
sensors = asyncio.run(
    read_all_sensors(
        engine,
        datetime.fromisoformat("2024-07-09"),
        datetime.fromisoformat("2024-07-10"),
    )
)
names = (
    [i.name + ", counts" for i in sensors["lidar_sensors"]]
    + [i.sensor + ", " + str(i.units) for i in sensors["ldp_sensors"]]
    + [
        i.sensor_name + " V" + str(i.meteostation_id) + ", " + i.units
        for i in sensors["meteo_sensors"]
    ]
    + [i + ", x10^3 cm^-3" for i in sensors["ions_sensors"]]
    + [i for i in sensors["dvo_meteo"]]
)

@app.callback(
    Output("names", "options"),
    Output("names", "value"),
    Output("date_picker", "max_date_allowed"),
    Input("plot_graphs", "n_clicks"),
    Input("date_range", "value"),
    Input("date_picker", "start_date"),
    Input("date_picker", "end_date"),
    State("names", "value"),
)
```

```
def set_names(_, date_range, start, end, values):
    if date_range:
        sensors = asyncio.run(
            read_all_sensors(
                engine,
                datetime.now() - timedelta(days=date_range),
                datetime.now(),
            )
        )
    else:
        if start and end:
            sensors = asyncio.run(
                read_all_sensors(
                    engine,
                    start,
                    end,
                )
            )
        else:
            return list(), list()
    names = (
        [i.name + ", counts" for i in sensors["lidar_sensors"]]
        + [i.sensor + ", " + str(i.units) for i in sensors["ldp_sensors"]]
        + [
            i.sensor_name + " V" + str(i.meteostation_id) + ", " + i.units
            for i in sensors["meteo_sensors"]
        ]
        + [i + ", x10^3 cm^-3" for i in sensors["ions_sensors"]]
        + [i for i in sensors["dvo_meteo"]]
    )
    val = [i for i in names if i in set(values)]
    return names, val, datetime.now().date() + timedelta(days=1)

@app.callback(
    Output("BNO_graph", "figure"),
    Output("DVO_graph", "figure"),
    Output("table", "data"),
    Input("interval-component", "n_intervals"),
    Input("plot_graphs", "n_clicks"),
    State("names", "value"),
    State("date_range", "value"),
    State("date_picker", "start_date"),
    State("date_picker", "end_date"),
)
def update_plot(_, n, names, date_range, start, end):
    if date_range:
        data = asyncio.run(
            read_all_data(
                engine,
                datetime.fromisoformat("2024-08-06 18:00") -
timedelta(days=date_range),
                datetime.fromisoformat("2024-08-06 18:00"),
            )
        )
        dvo_data = asyncio.run(
            read_dvo_data(
                None, datetime.now() - timedelta(days=date_range), datetime.now()
            )
        )
    else:
        data = asyncio.run(
```

```
            read_all_data(
                engine, datetime.fromisoformat(start), datetime.fromisoformat(end)
            )
        )
        dvo_data = asyncio.run(
            read_dvo_data(
                None, datetime.fromisoformat(start), datetime.fromisoformat(end)
            )
        )
    dvo_names = names[-6:]
    names = names[:-6]
    bno = plot.all_data_fig(data, names)
    dvo = plot.all_data_fig(dvo_data, dvo_names)
    res = [
        {
            "date": data[i.split(",")[0]]["data"]
            .index[-1]
            .strftime("%H:%M:%S %d %B %Y"),
            "sensor": i.split(",")[0],
            "status": "Работает"
            if datetime.now() - data[i.split(",")[0]]["data"].index[-1]
            < timedelta(minutes=30)
            else "Ошибка",
        }
        for i in names
    ]
    return (bno, dvo, res)

app.layout = html.Div(
    children=[
        html.H1(
            children="Лазерный мониторинг газов и аэрозолей в Баксанской
Нейтринной Обсерватории"
        ),
        dcc.Markdown(
            """
    В рамках научного проекта по поиску индикаторов
    предвестников землетрясений
    [РНФ грант №19-19-00712-П](https://rscf.ru/project/22-19-35084/),
    проводятся непрерывный мониторинг вариации аэрозолей,
    концентрации газов и метеопараметров в штольне
    [Баксанской нейтринной обсерватории.](https://www.inr.ru/bno/)
    Для мониторинга применяются высокочувствительный аэрозольный лидар
    и лидар дифференциального поглощения, разработанные
    в ФИЦ [«Институт Общей Физики им А.М. Прохрова РАН».](https://www.gpi.ru/)
                """
        ),
        dcc.RadioItems(
            options=[
                {"label": "1 день", "value": 1},
                {"label": "2 дня", "value": 2},
                {"label": "3 дня", "value": 3},
                {"label": "1 неделя", "value": 7},
                {"label": "2 недели", "value": 14},
                {"label": "1 месяц", "value": 30},
                {"label": "6 месяцев", "value": 183},
                {"label": "1 год", "value": 365},
                {"label": "Ввести диапазон", "value": False},
            ],
            value=3,
            inline=True,
```

```
            id="date_range",
        ),
        dcc.DatePickerRange(
            id="date_picker",
            min_date_allowed=date(2019, 8, 22),
            max_date_allowed=datetime.now().date() + timedelta(days=1),
            initial_visible_month=datetime.now().date(),
            start_date=datetime.now().date() - timedelta(weeks=1),
            end_date=datetime.now().date(),
            display_format="DD.MM.YYYY",
        ),
        html.Br(),
        dcc.Loading(
            html.Div(
                children=[
                    html.Label("Выбор графиков:"),
                    dcc.Dropdown(
                        options=names,
                        value=names,
                        multi=True,
                        id="names",
                        persistence=True,
                        maxHeight=500,
                    ),
                    html.Br(),
                    html.Button(
                        id="plot_graphs", n_clicks=0, children="Вывести график"
                    ),
                    html.Br(),
                ]
            )
        ),
        html.Div(
            children=[
                dcc.Loading(dcc.Graph(id="BNO_graph")),
                dcc.Loading(dcc.Graph(id="DVO_graph")),
            ],
            style={"display": "flex"},
        ),
        dash.dash_table.DataTable(
            id="table",
            columns=[
                {"name": "Сенсор", "id": "sensor"},
                {"name": "Последняя регистрация", "id": "date"},
                {"name": "Статус", "id": "status"},
            ],
            style_cell={"textAlign": "left"},
            style_as_list_view=True,
            style_data={
                "backgroundColor": "tomato",
                "color": "white",
                "whiteSpace": "normal",
                "height": "auto",
            },
            style_data_conditional=[
                {
                    "if": {"filter_query": '{status} = "Работает"'},
                    "backgroundColor": "green",
                }
            ],
        ),
```

```
        dcc.Interval(
            id="interval-component",
            interval=60 * 10 * 1000,  # in milliseconds
            n_intervals=0,
        ),
    ]
)
```

Листинг 7. Сбор данных с метеостанции

```
import asyncio
from datetime import datetime
from pathlib import Path

from bno_monitoring.sql_connect import Meteo_Data, Meteostations, Sensors
import pandas as pd
import typing
from sqlalchemy import Insert, select
from sqlalchemy.orm import Session
from bno_monitoring.bno_logger import log

async def read_meteo_data(
    file: Path,
) -> pd.DataFrame:
    """
    Чтение метеоданных из файла
    """
    try:
        meteo_data = pd.read_csv(
            file,
            sep="\t",
            header=0,
            encoding="cp1251",
            decimal=".",
            engine="python",
            na_values="--",
        )
        log.debug(f"Прочитаны метеоданные из файла {file}")
    except:
        raise
    return await set_datetime_index(meteo_data)

async def measurements(
    engine,
    file: Path,
    metestation: typing.Union[str, int],
    metestation_version: typing.Optional[str] = None,
) -> dict:
    data = await read_meteo_data(file)
    sql_meteostation = await select_meteostation(
        engine, metestation, metestation_version
    )
    sensors = await get_sensors_from_db(engine, sql_meteostation)
    columns_names = [sensor.sensor_name for sensor in sensors]
    data.columns = columns_names
    log.info(
        f"Подготовлено измерение. Метеостанция {sql_meteostation.name}
{sql_meteostation.version}.\n{data}"
    )
    return dict(meteostation=sql_meteostation, data=data)

async def set_datetime_index(data: pd.DataFrame) -> pd.DataFrame:
```

```
    """
    Расчет индекса datetime для данных где есть колнки Date и Time
    """
    data["Datetime"] = [
        datetime.strptime(d + " " + t, "%d-%m-%Y %H:%M:%S")
        for d, t in zip(data["Date"].values, data["Time"].values)
    ]
    data = data.drop(columns=["Date", "Time"])
    data = data.set_index("Datetime")
    log.debug("Расчитан индекс дата + время")
    return data

async def select_meteostation(
    engine,
    meteostation: typing.Union[str, int],
    meteostation_version: typing.Optional[str] = None,
) -> Meteostations:
    with Session(engine) as session:
        if isinstance(meteostation, int):
            sql_meteostation = session.scalar(
                select(Meteostations).where(Meteostations.id == meteostation)
            )
        else:
            if meteostation_version is not None:
                sql_meteostation = session.scalar(
                    select(Meteostations).where(
                        Meteostations.name == meteostation,
                        Meteostations.version == meteostation_version,
                    )
                )
            else:
                raise ValueError("lidar_version is required if lidar is str")
        if sql_meteostation is None:
            if isinstance(meteostation, str):
                if meteostation_version is None:
                    raise ValueError("lidar_version is required if lidar is str")
                sql_meteostation = Meteostations(
                    name=meteostation, version=meteostation_version
                )
            else:
                raise ValueError(f"No lidar in db with id {meteostation}")
            session.add(sql_meteostation)
            session.commit()
            session.refresh(sql_meteostation)
    log.debug(
        f"Метеостанция из БД. {sql_meteostation.id} {sql_meteostation.name}
{sql_meteostation.version}"
    )
    return sql_meteostation

async def get_sensors_from_db(
    engine,
    meteostation: Meteostations,
) -> typing.Sequence:
    with Session(engine) as session:
        sql_sensors = session.scalars(
            select(Sensors)
            .where(Sensors.meteostation_id == meteostation.id)
            .order_by(Sensors.sensor_position_in_file)
        ).all()
    log.debug(f"Сенсоры метеостанции. \n {sql_sensors}")
```

```
    return sql_sensors

async def add_data_to_db(
    engine,
    measurement: dict,
) -> None:
    with Session(engine) as session:
        for row in measurement["data"].itertuples():
            for item, sensor_name in zip(row[1:], measurement["data"].columns):
                session.execute(
                    Insert(Meteo_Data)
                    .values(
                        meteostation_id=measurement["meteostation"].id,
                        measurement_date=row[0],
                        sensor_name=sensor_name,
                        value=item,
                    )
                    .prefix_with("IGNORE")
                )
                log.info(
                    f"Добавление метеоданных в БД с игнором {row[0]}
{measurement['meteostation'].id} {sensor_name} {item}"
                )
                session.commit()

async def latest_measurement(engine, meteostation: Meteostations) -> datetime:
    with Session(engine) as session:
        measurement = session.scalar(
            select(Meteo_Data)
            .where(Meteo_Data.meteostation_id == meteostation.id)
            .order_by(Meteo_Data.measurement_date.desc())
            .limit(1)
        )
        if measurement is None:
            raise ValueError(
                f"No measurements in db for meteostation {meteostation.name}"
            )
        log.debug(f"Последнее измерение в БД {measurement.measurement_date}")
    return measurement.measurement_date

async def new_measurements(engine, measurement: dict) -> dict:
    measurement["data"] = measurement["data"][
        await latest_measurement(engine, measurement["meteostation"]) :
    ]
    return measurement

async def add_new_data_to_db(
    engine,
    file: Path,
    meteostation: typing.Union[int, str],
    meteostation_version: typing.Optional[str] = None,
):
    measurement = await new_measurements(
        engine, await measurements(engine, file, meteostation,
meteostation_version)
    )
    await add_data_to_db(engine, measurement)
```

Листинг 8. Чтение данных из SQL базы данных

```
from datetime import datetime, timedelta
from sqlalchemy import and_, create_engine, select
from sqlalchemy.orm import Session
from bno_monitoring.sql_connect import (
    Histogram,
    Ions_Signals,
    LDP_data,
    Lidars_Signals,
    Meteo_Data,
    Sensors,
)
import pandas

async def read_meteo_sensors(engine, start_date: datetime, end_date: datetime):
    with Session(engine) as session:
        sensors = session.execute(
            select(Meteo_Data.sensor_name, Meteo_Data.meteostation_id,
Sensors.units)
            .join(
                Sensors,
                and_(
                    Meteo_Data.sensor_name == Sensors.sensor_name,
                    Meteo_Data.meteostation_id == Sensors.meteostation_id,
                ),
            )
            .where(
                Meteo_Data.measurement_date >= start_date,
                Meteo_Data.measurement_date <= end_date,
            )
            .distinct()
        )
    return sensors

async def read_meteo_data(engine, start_date: datetime, end_date: datetime):
    sensors = await read_meteo_sensors(engine, start_date, end_date)
    result = {}
    for sensor in sensors:
        data = pandas.read_sql(
            str(
                select(Meteo_Data.measurement_date, Meteo_Data.value)
                .where(
                    Meteo_Data.sensor_name == sensor.sensor_name,
                    Meteo_Data.meteostation_id == sensor.meteostation_id,
                    Meteo_Data.measurement_date >= start_date,
                    Meteo_Data.measurement_date <= end_date,
                )
                .order_by(Meteo_Data.measurement_date.asc())
                .compile(engine, compile_kwargs={"literal_binds": True})
            ),
            engine,
            "measurement_date",
            parse_dates=True,
        )
        result[sensor.sensor_name + " V" + str(sensor.meteostation_id)] = {
            "data": data,
            "units": sensor.units,
        }
    return result

async def read_ldp_sensors(engine, start_date: datetime, end_date: datetime):
    with Session(engine) as session:
```

```
        sensors = session.execute(
            select(LDP_data.sensor, LDP_data.units)
            .where(
                LDP_data.measurement_date >= start_date,
                LDP_data.measurement_date <= end_date,
            )
            .distinct()
        )
    return sensors

async def read_ldp_data(engine, start_date: datetime, end_date: datetime):
    result = {}
    sensors = await read_ldp_sensors(engine, start_date, end_date)
    for sensor in sensors:
        data = pandas.read_sql(
            str(
                select(LDP_data.measurement_date, LDP_data.value)
                .where(
                    LDP_data.sensor == sensor.sensor,
                    LDP_data.units == sensor.units,
                    LDP_data.measurement_date >= start_date,
                    LDP_data.measurement_date <= end_date,
                )
                .order_by(LDP_data.measurement_date.asc())
                .compile(engine, compile_kwargs={"literal_binds": True})
            ),
            engine,
            "measurement_date",
            parse_dates=True,
        )
        result[sensor.sensor] = {"data": data, "units": sensor.units}
    return result

async def read_lidar_sensors(engine, start_date: datetime, end_date: datetime):
    with Session(engine) as session:
        sensors = session.execute(
            select(Lidars_Signals.name)
            .join(Histogram, Lidars_Signals.histogram_id == Histogram.id)
            .where(
                Histogram.measurement_date >= start_date,
                Histogram.measurement_date <= end_date,
            )
            .distinct()
        )
    return sensors

async def read_lidar_signals(engine, start_date: datetime, end_date: datetime):
    result = {}
    sensors = await read_lidar_sensors(engine, start_date, end_date)
    if sensors:
        for sensor in sensors:
            data = pandas.read_sql(
                str(
                    select(Histogram.measurement_date, Lidars_Signals.value)
                    .join(Lidars_Signals, Histogram.id ==
Lidars_Signals.histogram_id)
                    .where(
                        Lidars_Signals.name == sensor.name,
                        Histogram.measurement_date >= start_date,
                        Histogram.measurement_date <= end_date,
                    )
```

```
                    .order_by(Histogram.measurement_date.asc())
                    .compile(engine, compile_kwargs={"literal_binds": True})
                ),
                engine,
                "measurement_date",
            )
            result[sensor.name] = {"data": data, "units": "counts"}
        return result

async def read_ions_sensors():
    sensors = ["I-", "I+"]
    return sensors


async def read_all_sensors(engine, start_date: datetime, end_date: datetime):
    lidar_sensors = await read_lidar_sensors(engine, start_date, end_date)
    ldp_sensors = await read_ldp_sensors(engine, start_date, end_date)
    meteo_sensors = await read_meteo_sensors(engine, start_date, end_date)
    ions_sensors = await read_ions_sensors()
    dvo_meteo = await read_meteo_data(create_engine("mysql+mysqldb:// "),
start_date+timedelta(hours=9), end_date+timedelta(hours=9))
    return {
        "lidar_sensors": lidar_sensors,
        "ldp_sensors": ldp_sensors,
        "meteo_sensors": meteo_sensors,
        "ions_sensors": ions_sensors,
        "dvo_meteo": dvo_meteo,
    }

async def read_ions_data(engine, start_date: datetime, end_date: datetime):
    data_ions_minus = pandas.read_sql(
        f"select Ions_Signals.measurement_datetime, Ions_Signals.ions_minus from
Ions_Signals where Ions_Signals.measurement_datetime > '{start_date +
timedelta(hours=9)}' and Ions_Signals.measurement_datetime  <
'{end_date+timedelta(hours=9)}';",
        engine,
        "measurement_datetime",
    )
    data_ions_minus.columns = ["value"]
    data_ions_minus.index = [i - timedelta(hours=9) for i in
data_ions_minus.index]
    data_ions_plus = pandas.read_sql(
        f"select Ions_Signals.measurement_datetime, Ions_Signals.ions_plus from
Ions_Signals where Ions_Signals.measurement_datetime > '{start_date +
timedelta(hours=9)}' and Ions_Signals.measurement_datetime  <
'{end_date+timedelta(hours=9)}';",
        engine,
        "measurement_datetime",
    )
    data_ions_plus.columns = ["value"]
    data_ions_plus.index = [i - timedelta(hours=9) for i in data_ions_plus.index]
    result = {"I-": {"data": data_ions_minus}, "I+": {"data": data_ions_plus}}
    return result

async def read_dvo_data(engine, start_date: datetime, end_date:datetime):
    engine = "mysql+mysqldb:// "
    meteo = await read_meteo_data(create_engine(engine),
start_date+timedelta(hours=9), end_date+timedelta(hours=9))
    ions_data = await read_ions_data(engine, start_date, end_date)
    for sensor in meteo.keys():
        data = meteo[sensor]['data']
```

```
        data.index = [i - timedelta(hours=9) for i in data.index]
        meteo.update({sensor:{'data': data, 'units':meteo[sensor]['units']}})
    all_data = {}
    for d in [meteo, ions_data]:
        all_data.update(d)
    return all_data

async def read_all_data(engine, start_date: datetime, end_date: datetime):
    meteo = await read_meteo_data(engine, start_date, end_date)
    ldp = await read_ldp_data(engine, start_date, end_date)
    lidar_data = await read_lidar_signals(engine, start_date, end_date)
    all_data = {}
    for d in [lidar_data, ldp, meteo]:
        all_data.update(d)
    return all_data
```

Листинг 9. Формат базы данных SQL

```
from typing import Optional
from datetime import datetime
from sqlalchemy import (
    ForeignKey,
    ForeignKeyConstraint,
    create_engine,
    Integer,
    String,
    DateTime,
    Float,
)
from sqlalchemy.dialects.mysql import insert
from sqlalchemy.orm import DeclarativeBase, Mapped, mapped_column

engine = create_engine("mysql+mysqldb:// ")


class lidars_database(DeclarativeBase):
    pass


class Lidar(lidars_database):
    __tablename__ = "Lidars"
    id: Mapped[int] = mapped_column(
        primary_key=True, autoincrement=True, unique=True, nullable=False
    )
    name: Mapped[str] = mapped_column(String(50))
    version: Mapped[str] = mapped_column(String(10))
    creation_date: Mapped[datetime] = mapped_column(DateTime)


class Histogram(lidars_database):
    __tablename__ = "Histograms"
    id: Mapped[int] = mapped_column(
        primary_key=True, autoincrement=True, unique=True, nullable=False
    )
    lidar_id: Mapped[int] = mapped_column(
        ForeignKey("Lidars.id", ondelete="CASCADE", onupdate="CASCADE"),
nullable=False
    )
    measurement_date: Mapped[datetime] = mapped_column(DateTime, nullable=False)
    pulses: Mapped[int] = mapped_column(Integer, nullable=False)
```

```
    channel: Mapped[Optional[str]] = mapped_column(String(50))
    parameters: Mapped[Optional[str]] = mapped_column(String(100))


class Data(lidars_database):
    __tablename__ = "Data"
    histogram_id: Mapped[int] = mapped_column(
        ForeignKey("Histograms.id", ondelete="CASCADE", onupdate="CASCADE"),
        primary_key=True,
        nullable=False,
    )
    distance: Mapped[int] = mapped_column(Integer, primary_key=True,
nullable=False)
    units: Mapped[str] = mapped_column(String(10), nullable=False)
    signals: Mapped[int] = mapped_column(Integer, nullable=False)


class Lidars_Signals(lidars_database):
    __tablename__ = "Lidars_Signals"
    histogram_id: Mapped[int] = mapped_column(
        ForeignKey("Histograms.id", ondelete="CASCADE", onupdate="CASCADE"),
        primary_key=True,
        nullable=False,
    )
    name: Mapped[str] = mapped_column(String(50), primary_key=True,
nullable=False)
    start: Mapped[int] = mapped_column(Integer, nullable=False)
    stop: Mapped[int] = mapped_column(Integer, nullable=False)
    value: Mapped[int] = mapped_column(Integer, nullable=False)


class Meteostations(lidars_database):
    __tablename__ = "Meteostations"
    id: Mapped[int] = mapped_column(
        primary_key=True, autoincrement=True, unique=True, nullable=False
    )
    name: Mapped[str] = mapped_column(String(50))
    version: Mapped[str] = mapped_column(String(10))
    creation_date: Mapped[datetime] = mapped_column(DateTime)


class Sensors(lidars_database):
    __tablename__ = "Sensors"
    meteostation_id: Mapped[int] = mapped_column(
        ForeignKey("Meteostations.id", ondelete="CASCADE"),
        primary_key=True,
        nullable=False,
    )
    sensor_name: Mapped[str] = mapped_column(
        String(50), nullable=False, primary_key=True
    )
    sensor_position_in_file: Mapped[int] = mapped_column(Integer)
    units: Mapped[str] = mapped_column(String(10), nullable=False)


class Meteo_Data(lidars_database):
    __tablename__ = "Meteo_Data"
    __table_args__ = (
        ForeignKeyConstraint(
            ["meteostation_id", "sensor_name"],
            ["Sensors.meteostation_id", "Sensors.sensor_name"],
```

```
            ondelete="CASCADE",
            onupdate="CASCADE",
        ),
    )
    meteostation_id: Mapped[int] = mapped_column(
        Integer, primary_key=True, nullable=False
    )
    sensor_name: Mapped[str] = mapped_column(
        String(50), primary_key=True, nullable=False
    )
    measurement_date: Mapped[datetime] = mapped_column(
        DateTime, nullable=False, primary_key=True
    )
    value: Mapped[float] = mapped_column(Float, nullable=False)
class LDP_data(lidars_database):
    __tablename__ = "LDP_data"
    measurement_date: Mapped[datetime] = mapped_column(
        DateTime, primary_key=True, nullable=False
    )
    sensor: Mapped[str] = mapped_column(String(50), primary_key=True,
nullable=False)
    value: Mapped[float] = mapped_column(Float)
    units: Mapped[Optional[str]] = mapped_column(String(50))


class Ions_Signals(lidars_database):
    __tablename__ = "Ions_Signals"
    measurement_datetime: Mapped[datetime] = mapped_column(
        DateTime, nullable=False, primary_key=True
    )
    ions_minus: Mapped[float] = mapped_column(Float, nullable=False)
    ions_plus: Mapped[float] = mapped_column(Float, nullable=False)


def check_insert(table, conn, keys, data_iter):
    data = [dict(zip(keys, row)) for row in data_iter]
    stmt = insert(table.table).values(data).prefix_with("IGNORE")
    result = conn.execute(stmt)
    return result.rowcount
```