ВВЕДЕНИЕ

Дистанционное зондирование Земли – это способ получения информации об объекте без непосредственного физического контакта с ним. На борту летательного аппарата (например, спутника либо самолёта) устанавливается спектрометр, задачей которого является фиксация излучения с поверхности, затем бортовая система осуществляет предобработку полученных данных и передаёт их в центр приёма информации. При этом в зависимости от типа спектрометра рабочий диапазон длин волн, может составлять от долей микрометра до метров.

В зависимости от типа спектрометра различают мультиспектральные и гиперспектральные данные. Основное отличие в том, что гиперспектральные спектрометры фиксируют данные в виде непрерывного диапазона спектра с определённым шагом, в то время как мультиспектральные данные могут иметь такое же количество каналов данных, но распределенных в спектральном диапазоне неравномерно.

Тенденции развития дистанционного зондирования показывают, что акцент в исследованиях смещается в область гиперспектральной съёмки. Однако существует несколько факторов препятствующих их широкому распространению: отсутствие достаточного количества воздушных судов, оборудованных соответствующими спектрометрами; проблемы связанные с обработкой и интерпретацией больших потоков информации, формируемых этими приборами. В связи с этим особенно остро стоит вопрос о создании спутниковой гиперспектральной аппаратуры и технологий обработки получаемой с помощью нее информации на борту летательного аппарата.

Данные, передаваемые с летательного аппарата в центр приёма, представляют собой трёхмерный куб, характеризующийся следующими разрешениями: пространственным (определяет площадь поверхности); спектральным (определяет охватываемый спектральный диапазон); радиометрическим (определяет число уровней сигнала, которые сенсор может зарегистрировать).

Подобная структура, с учётом непрерывного спектрального диапазона, приводит к формированию существенного объёма данных передаваемых на Землю и актуализирует задачу сжатия. Например, данные спектрометра AVIRIS, которые используются для разработки алгоритмов и программного обеспечения для обработки гиперспектральных снимков, имеют следующие характеристики: ширина изображения 677 пикселов, 224 спектральных канала, 12 бит на канал, что в общем случае приводит к 222,1 Кб данных на строку. С учётом характеристик современных радиоканалов связи и того, что съёмка ведётся без остановки, важнейшими требованиями к алгоритмам сжатия данных являются высокий коэффициента сжатия данных и низкие требования к вычислительной мощности алгоритма, что связано с техническими ограничениями летательных аппаратов.

Цель данного дипломного проекта – выбрать наиболее подходящий алгоритм и разработать программный комплекс для сжатия гиперспектральных данных, который должен иметь высокий коэффициент сжатия данных при минимальных потерях качества, а также обладать низкой вычислительной сложностью, простотой в использовании.

В соответствии с поставленной целью были поставлены следующие задачи:

- обзор существующих алгоритмов сжатия;
- разработка структурной схемы системы
- выбор алгоритма подходящего по требованиям;
- реализация программного комплекса;
- реализация модулей поддержки и визуализации;

1 ОБЗОР ЛИТЕРАТУРЫ

1.1 Обзор аналогов

Гиперспектральное изображение - это трехмерный массив данных (куб данных), который включает в себя пространственную информация (2D) об объекте, дополненную спектральной информацией (1D) по каждой пространственной координате. Иными словами, каждой точке изображения соответствует спектр, полученный в этой точке снимаемого объекта.

До появления техники записи гиперспектральных изображений, для получения информации об объекте (участке местности) использовались мультиспектральные изображения, то есть наборы фотографий, полученные с помощью цветных светофильтров. Изначально такой подход, использующий пространственную картину спектрального распределения, применялся лишь для дистанционного зондирования окружающей среды и распознавания боевых целей. Однако, в настоящее время он широко используется в области биомедицинской оптики. Гиперспектральные визоры вскоре станут одной из основных технологий в медицинской сфере.

Выделяют два подхода к сжатию гиперспектральных данных: с применением общеизвестных методик сжатия и с адаптацией алгоритма под заданные условия.

В рамках первого подхода чаще всего используются алгоритмы сжатия без потерь и почти без потерь (когда потери информации не превышают уровень шума, вносимого используемым спектрометром). Они разделяются на следующие основные классы, главное различие которых сводится к аппаратным ресурсам: на основе предсказания (linear prediction(LP), fast lossless(FL), spectral oriented least squares (SLSQ), correlation-based condition average prediction(CCAP), M-CALIC); поиск по таблице (lookup table (LUT), locally averaged interband scalling lookup tables (LAIS-LUT)); вейвлеты (3D-SPECK, Dual Tree BEZW, 3D-SPIHT).

В алгоритмах сжатия, имеющих в основе алгоритмов предсказания выделяется некоторая окрестность, над которой выполняется математическое действие(предсказание). Результат предсказания вычитается из оригинального значения и формируется ошибка предсказания, которая передаётся в блок энтропийного кодирования, результатом которого является сжатый поток данных. Восстановление осуществляется в обратной последовательности. В качестве алгоритма кодирования могут использоваться, например, коды Голомба-Райса или любой арифметический кодек, допускающий аппаратную реализацию.

Основным недостатком многих алгоритмов предсказания является высокая вычислительная нагрузка при небольшом использовании оперативной памяти.

Алгоритмы на основе поиска по таблице обеспечивают ускорение процесса вычисления, основанное на существенности корреляции между спектральными каналами. Размерность таблицы – число спектральных каналов, умноженное на максимально допустимое значение при данном радиометрическом разрешении. По текущему значению пиксела делается запрос в таблицу, и возвращаемое значение считается предсказанным. Дальнейшая обработка эквивалентна алгоритмам предсказания.

Алгоритмы на основе дискретного вейвлет-преобразования являются наиболее требовательными ко всем вычислительным ресурсам. Данный класс предполагает предварительный перевод спектральной плоскости в частотную область. После этого возможно организовать обработку таким образом, чтобы система кодировала в первую очередь наиболее значимые вейвлет-коэффициенты, постепенно смещаясь в область с наименее значимыми коэффициентами. Такой подход позволяет реализовать как сжатие без потерь (при обработке всех вейвлет-коэффициентов), так и управляемое сжатие с потерями. Главный недостаток алгоритмов этого класса – вычислительная сложность, связанная с преобразованием в частотную область куба данных и требования к пропускной способности памяти из-за случайных переходов в памяти от низкочастотных к высокочастотным вейвлет-коэффициентам.

Второй подход основан на существенной избыточности получаемых данных и связан с большим спектральным разрешением. Алгоритмы данного класса основываются на следующих упрощениях: 1) условия съемки заведомо известны; при таком подходе появляется возможность на борту летательного аппарата удалить неинформативные каналы (например, учесть влияние атмосферы) либо, наоборот, выделить наиболее информативные, т.е. в любом случае получить мультиспектральные данные;

2) выполнение полного либо частичного анализа полученных данных и передача результата, а не самих данных.

Достоинством алгоритмов, реализующих данный подход является передача только необходимых данных и существенное понижение объёма передаваемых данных. Тем не менее алгоритмы практически не реализуемы на борту летательного аппарата из-за их вычислительной сложности.

1.1.1 Алгоритм сжатия 3D-SPECK

Алгоритм 3D-SPECK (three dimensional Set Partitioning Embedded bloCK) это алгоритм сжатия без потерь (lossless). Трехмерное вейвлет-преобразование использует межполосную корреляцию объёмных блоков изображения. Трёхмерная структура этого алгоритма включает использование межполосных зависимостей и корреляций. Так же алгоритм 3D-SPECK поддерживает расщепление блоков для сортировки важных пикселов, то есть если блок кода содержит важные коэффициенты он разделяется на небольшие подблоки. Также,

если гиперспектральное изображение содержит концентрацию энергии в высокочастотных каналах, алгоритм должен хорошо справиться с таким изображением.

3D-SPECK включает два связанных списка: список незначительных наборов и список значимых пикселов. Процесс обработки состоит из четырёх этапов: этапа инициализации, сортировки, уточнения и квантования. Во время этапа сортировки метод разделения блоков принимается после теста на значимость. На этапе уточнения некоторые коэффициенты передаются и процесс квантование продолжается для следующей уменьшенной битовой плоскости, пока не будет достигнута приемлемая скорость передачи данных. Преобразованный коэффициент трёхмерного дискретного вейвлет-преобразования декоррелирует пространственные и спектральные компоненты гиперспектрального изображения. Это выявляет некоторую избыточность, которая может быть использована во время процесса кодирования. Кодировщик следует за основным алгоритмом сортировки, как в 3D-SPECK. Поэтому межзональную зависимость можно использовать автоматически. Для того чтобы сопоставить трёхмерные коэффициенты с одномерным массивом, используется рекурсивная Z-последовательность или кривая Мортона. Кривая Мортона – это функция которая отображает многомерные данные в одномерные, сохраняя локальность точек данных.

После линейной системы индексирования количество коэффициентов сохраняется в одном массиве длинной I, где I = строки * столбцы * фреймы, и является массивом величин. Массив таблицы состояний, основанный на линейной индексации, также имеет длину I с 4 битами на коэффициент, который является отмеченной частью. Между величиной и меткой существует взаимно однозначное соответствие. В этом алгоритме так же устранена проблема вейвлет-преобразований связанная с избыточным потреблением памяти.

1.1.2 Алгоритм сжатия 3D-SPIHT

Алгоритм сжатия 3D-SPIHT имеет в основе дискретное вейвлет преобразование. Степень сжатия при использовании данного алгоритма сильно зависит от самого изображения(снимаемая территория, наличие шумов).

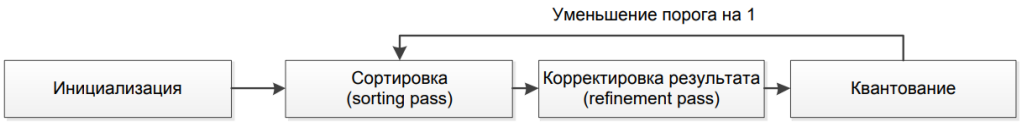

Рисунок 1.1 – Общий вид алгоритма 3D-SPIHT [1]

На этапе инициализации в список несущественных наборов (LIS) помещаются коэффициенты от вейвлет-преобразования подмножеств изображения для текущего уровня декомпозиции. После этого осуществляется анализ элементов в наборе LIS и сравнение результата с пороговым значением n . При

прохождении порогового значения осуществляется разбиение анализируемого элемента на 8 эквивалентных подмножеств и процедура сортировки повторяется до тех пор, пока не будет найдено достоверное значение пикселя из оригинального множества. Достоверное значение пикселя переносится в массив LSP (список существенных пикселей), а из множества LIS удаляется. После окончания сортировки осуществляется дополнительная корректировка и квантование данных. Дополнительная корректировка в зависимости от реализации включает обработку текущего пикселя и сравнения его с текущим пороговым значением. В результате этого сравнения принимается решение о формировании соответствующего набора битов в выходной поток. После этого пороговое значение n уменьшается на 1 и осуществляется возврат на этап сортировки. Полученный массив данных после работы алгоритма является сжатым гиперспектральным изображением. Общий вид алгоритма представлен выше (см. рисунок 1.1)

1.1.2 Алгоритм сжатия 3D-BEZW

Алгоритм сжатия 3D-BEZW это алгоритм сжатия изображения с потерями. При низких скоростях передачи, то есть при высоких коэффициентах сжатия, большинство коэффициентов, создаваемых преобразование поддиапазона (например, вейвлет-преобразование), будет равно нулю или очень близка к нулю. Это происходит потому что изображения «реального мира», как правило, содержат в основном низкочастотную информацию (высоко коррелируют). Однако там, где находится высокочастотная информация (например, грани на изображении), это особенно важно для человека с точки зрения восприятия качества картинки, и поэтому должно быть точно представлено в любой высококачественной схеме кодирования. Рассматривая преобразованные коэффициенты как дерево (или деревья) с наименьшими частотными коэффициентами в корневом узле и с дочерними элементами каждого узла дерева являются пространственно связанными коэффициентами в следующем более высокочастотном поддиапазоне, существует высокая вероятность того, что один или несколько поддеревья будут состоять полностью из коэффициентов, которые равны нулю или почти равны нулю, такие поддеревья называются нулевыми. В связи с этим мы используем термины узел и коэффициент взаимозаменяемо, а когда мы обращаемся к детям коэффициента, мы имеем в виду дочерние коэффициенты узла в дереве, где этот коэффициент расположен. Мы используем дочерние элементы, чтобы ссылаться на непосредственно связанные узлы ниже в дереве и потомках, чтобы ссылаться на все узлы, которые находятся ниже определенного узла в дереве, даже если они не связаны напрямую. В схеме сжатия изображений на основе нулевого дерева, такой как EZW и SPIHT, целью является использование статистических свойств деревьев для эффективного кодирования местоположений значимых коэффициентов. Поскольку большинство коэффициентов будут равны нулю или близки к нулю, пространственные местоположения значимых коэффициентов составляют

значительную часть общего размера типичного сжатого изображения. Коэффициент (аналогично дереву) считается значимым, если его величина (или величины узла и всех его потомков в случае дерева) превышает определенный порог. Начав с порога, близкого к максимальным значениям коэффициента и итеративно его уменьшая, можно создать сжатое представление изображения, которое постепенно добавляет более мелкие детали. Из-за структуры деревьев весьма вероятно, что если коэффициент в конкретной полосе частот незначителен, то все его потомки (пространственно связанные коэффициенты более высокой полосы частот) также будут значительными. EZW использует четыре символа для представления (а) корень нулей, (б) изолированный ноль (коэффициент, который незначителен, но имеющий значительные потомки), (в) значительный положительный коэффициент и (г) значительный отрицательный коэффициент. Таким образом, символы могут быть представлены двумя двоичными битами. Алгоритм сжатия состоит из нескольких итераций через доминирующий проход и подчиненный проход, порог обновляется (уменьшается в два раза) после каждой итерации. Доминирующий проход кодирует значение коэффициентов, которые еще не были признаны значимыми в предыдущих итерациях, путем сканирования деревьев и испускания одного из четырех символов. Дети коэффициента только сканируются, если коэффициент был признан значимым, или если коэффициент был изолированным нулем. Субординированный проход испускает один бит (самый старший бит каждого коэффициента, который до сих пор не испускался) для каждого коэффициента, который был признан значительным в предыдущих пропусков значимости. Таким образом, подчиненный проход аналогичен кодированию битовой плоскости. Есть несколько важных особенностей, которые следует отметить. Во-первых, в любой момент можно остановить алгоритм сжатия и получить приближение исходного изображения, чем больше количество полученных бит, тем лучше изображение. Во-вторых, из-за того, что алгоритм сжатия структурирован как ряд решений, один и тот же алгоритм может быть запущен в декодере для восстановления коэффициентов, но с решениями, принимаемыми в соответствии с входящим потоком битов. В практических реализациях было бы обычно использовать энтропийный код, такой как арифметический код, для дальнейшего улучшения производительности доминирующего прохода. Биты из подчиненного прохода обычно являются случайными, что энтропийное кодирование не дает дополнительного усиления кодирования. Алгоритм, основанный на преобразовании, сканирует преобразованное изображение несколько раз и кодирует значимые коэффициенты по отношению к нескольким пороговым значениям. Алгоритм эффективен, поскольку он основан на неперекрывающихся асимметричных древовидных структурах, которые интерполируют отношения между вейвлет-коэффициентами в разных масштабах поддиапазонов. Естественно, дерево должно быть спроектировано в соответствии со свойствами трансформированного изображения, так что ветви дерева к бо-

лее высокочастотным поддиапазонам происходят в одной и той же пространственной ориентации; другими словами, коэффициенты корней должны быть увеличены из низкочастотных поддиапазонов.

1.2 Обзор AVIRIS

AVIRIS (Airborn Visible and InfraRed Imaging Spectrometer) – бортовой спектрометр видимого и инфракрасного диапазонов.

AVIRIS - это проверенный инструмент в области дистанционного зондирования Земли. Это уникальный оптический датчик, который обеспечивает калиброванные изображения спектрального излучения в 224 смежных спектральных каналах (диапазонах) с длиной волны от 400 до 2500 нанометров. AVIRIS установлен на четырёх авиационных платформах: самолет ER-2 NASA, турбовинтовой двигатель Twin Otter International, Proteus Scaled Composites и WB-57 НАСА. ER-2 летит примерно на 20 км над уровнем моря, около 730 км / час. Самолет Twin Otter летает на высоте 4 км над уровнем земли со скоростью 130 км / час. AVIRIS пролетел в Северной Америке, Европе, частях Южной Америки и Аргентины.

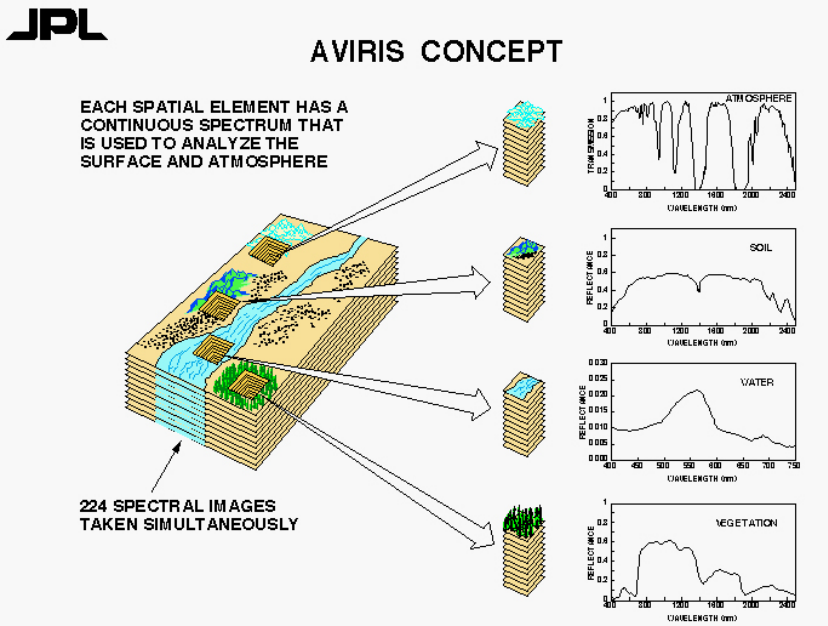

Рисунок 1.2 – Иллюстрация измерительных возможностей прибора AVIRIS[2]

Основной целью проекта AVIRIS является выявление, измерение и мониторинг составляющих земной поверхности и атмосферы на основе сигналов молекулярного поглощения и рассеяния частиц. Исследования с данными AVIRIS в основном сосредоточены на понимании процессов, связанных с глобальной окружающей средой и изменением климата.

Инструмент AVIRIS содержит 224 (см. рисунок 1.2) различных детекторов, каждый из которых имеет диапазон чувствительности к длине волны (также известный как спектральная пропускная способность), приблизительно 10 нанометров (нм), что позволяет покрывать весь диапазон между 380 нм и 2500 нм. Когда данные из каждого детектора изображаются на графике, он дает полный спектр VIS-NIR-SWIR. Сравнение полученного спектра с показателями известных веществ показывает информацию о составе области, рассматриваемой прибором.

Характеристики спектрометра AVIRIS

– Скорость передачи данных 17 Мбит/с до 1994 года, 20,4 Мбит/с с 1995 по 2004 год, 16 бит с 2005 года;
– 10-битная кодировка данных в 1994 году, 12-битная с 1995 года;
– Скорость сканирования 12 Гц;
– Детекторы охлаждаемые жидким азотом;
– 10 нанометров номинальная пропускная способность канала, калиброванная с точностью до 1 нанометра;
– 34 градуса общего поля зрения;
– Мгновенное поле зрения 1 миллирадиан, калиброванный с точностью до 0,1 мрад
– Сканирование «Whisk broom»
– 76 Гигабайт на носителе для записи;
– Кремниевые детекторы для видимого диапазона

Ниже приведен пример спектра одного пиксела от AVIRIS (см. рисунок 1.3). Ось X представляет собой длину волны канала в микрометрах, также известную как микрон (один микрон = 1000 нанометров). Ось y - это сияние, обычно выраженное в единицах микроватт на квадратный сантиметр на нанометр на стерадиан.

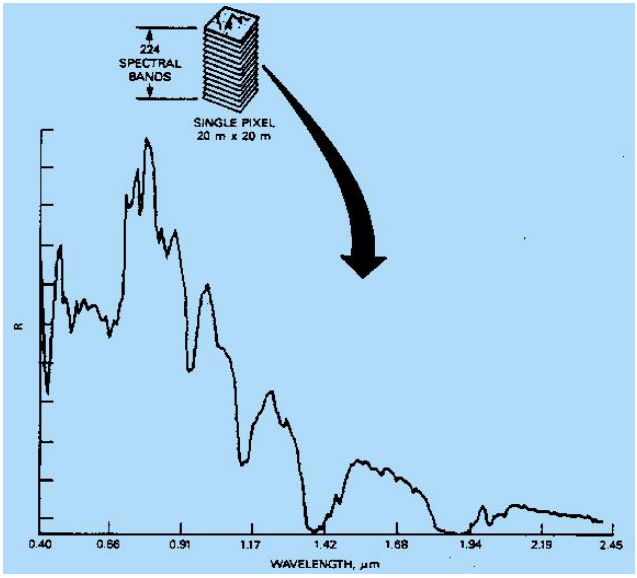

Рисунок 1.3 – Пример спектра одного пиксела от AVIRIS [3]

1.3 Обзор языка программирования C++

Для реализации данного дипломного проекта, из-за поставленных задач был выбран язык программирования C++.

Си++ (англ. C++) — компилируемый строго типизированный язык программирования общего назначения. Поддерживает разные парадигмы программирования: процедурную, обобщённую, функциональную; наибольшее внимание уделено поддержке объектно-ориентированного программирования.

Своими корнями он уходит в язык Си, который был разработан в 1969—1973 годах в компании Bell Labs программистом Деннисом Ритчи (Dennis Ritchie). В начале 1980-х годов датский программист Бьерн Страуструп (Bjarne Stroustrup), который в то время работал в компании Bell Labs, разработал C++ как расширение к языку Си. Фактически вначале C++ просто дополнял язык Си некоторыми возможностями объектно-ориентированного программирования. И поэтому сам Страуструп вначале называл его как "C with classes" ("Си с классами").

C++ является мощным языком, унаследовав от Си богатые возможности по работе с памятью. Поэтому нередко C++ находит свое применение в системном программировании, в частности, при создании операционных систем, драйверов, различных утилит, антивирусов и т.д. К слову сказать, ОС Windows большей частью написана на C++. Но только системным программированием применение данного языка не ограничивается. C++ можно использовать в программах любого уровня, где важны скорость работы и производительность. Нередко он применяется для создания графических приложений, различных прикладных программ. Также особенно часто его используют для создания игр с богатой насыщенной визуализацией. Кроме того, в последнее время набирает ход мобильное направление, где C++ тоже нашел свое применение. И даже в веб-разработке также можно использовать C++ для создания веб-приложений или каких-то вспомогательных сервисов, которые обслуживают веб-приложения. В общем C++ - язык широкого пользования, на котором можно создавать практически любые виды программ.

C++ является компилируемым языком, а это значит, что компилятор транслирует исходный код на C++ в исполняемый файл, который содержит набор машинных инструкций. Но разные платформы имеют свои особенности, поэтому скомпилированные программы нельзя просто перенести с одной платформы на другую и там уже запустить. Однако на уровне исходного кода программы на C++ по большей степени обладают переносимостью, если не используются какие-то специфичные для текущей ос функции. А наличие компиляторов, библиотек и инструментов разработки почти под все распространенные платформы позволяет компилировать один и тот же исходный код на C++ в приложения под эти платформы.

2. СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ

Изучив теоретические аспекты разрабатываемой системы и выработав список требований необходимых для разработки системы, разбиваем систему на функциональные блоки(модули). Это необходимо для обеспечения гибкой архитектуры. Такой подход позволяет изменять или заменять модули без изменения всей системы в целом.

В разрабатываемом программном комплексе можно выделить следующие функциональные блоки:

- блок визуализации
- блок работы с файловой системой
- блок управления
- блок увеличения битовой глубины
- блок спектрального преобразования
- блок уменьшения битовой глубины
- блок двумерного кодирования промежуточных изображений
- блок кодирования конечного изображения

Структурная схема, иллюстрирующая перечисленные блоки и связи между ними приведена на чертеже ГУИР.400201.081 С1.

Каждый модуль выполняет свою задачу. Чтобы система работала каждый модуль взаимодействует с другими модулями путем обмена данными, используя различные форматы и протоколы.

В данном программном комплексе будет реализовано сжатие гиперспектральных данных по стандарту CCSDS 112.1-B-1. Соответственно, для большей части проекта использование сторонних библиотек недопустимо, вследствие чего весь основной функционал будет реализован самостоятельно. Из-за высоких требований к скорости обработки проекта и эффективности сжатия было решено использовать язык программирования C++ как основной, т.к. он обеспечивает наименьшие потери производительности при достаточном функционале.

Рассмотрим функциональные блоки программного комплекса.

Блок визуализации содержит функции и данные для визуализации и построения диаграмм для операций над исходным гиперспектральным изображением. Для реализации этой компоненты будет использоваться кроссплатформенный набор инструментов Qt-toolkit. Эта библиотека задумывалась и начиналась как набор инструментов для быстрой разработки графических интерфейсов приложений на языке C++, с целью упростить жизнь программистов, пишущих на C++ кроссплатформенные, переносимые GUI приложения, которые должны работать и в среде Windows и в среде Unix, Linux.

Важным преимуществом Qt является хорошо продуманный, логичный и стройный набор классов, предоставляющий очень высокий уровень абстракции. Благодаря использованию Qt, приходится писать значительно меньше кода, чем это имеет место при использовании, например, библиотеки классов

MFC. Сам же код выглядит стройнее и проще, логичнее и понятнее, чем аналогичный по функциональности код MFC или код, написанный с использованием «родного» для X11 тулкита Xt. Его легче поддерживать и развивать.

Кроме того, даже если в данный конкретный момент для реализации задуманного приложения не нужна кроссплатформенность, никто не может знать, что понадобится завтра. В случае использования Qt для того чтобы адаптировать исходное приложение для работы в другой операционной системе понадобится всего лишь перекомпиляция исходного кода. В случае же использования, например, MFC или «родных» системных API понадобится много тяжёлой работы по портированию, адаптации и отладке, а то и переписыванию с нуля существующего исходного кода для другой ОС или аппаратной платформы.

Блок работы с файловой системой является блоком необходимым для взаимодействия программного комплекса и файловой системы. Для того чтобы подать на вход программы необходимые данные и для сохранения конечного результата необходимо реализовать блок работы с файловой системой. Так как язык программирования, выбранный для реализации проекта содержит более чем исчерпывающий функционал для работы с файловой системой, было решено не использовать сторонних фреймворков или библиотек.

Блок управления предназначен для взаимодействия пользователя, пользовательского интерфейса и остальных блоков программного комплекса и будет реализован с использованием C++ и библиотеки Qt. Это наиболее подходящий для данной задачи язык программирования так как он блестяще справляется с реализацией подобного функционала, при этом расширение возможностей данного блока в будущем не станет проблемой.

Блок увеличения битовой глубины реализует этап повышения битовой частоты, применяемый к изображению перед этапом спектрального преобразования. На этом этапе частота изображения повышается путём добавления дополнительных битов в младшие разряды чисел. Целью этого этапа является повышение битовой глубины входного изображения до максимальной, поддерживаемой данной реализацией этапа преобразования. Эта увеличенная битовая глубина обычно обеспечивает более высокую производительность кодирования при высокоточном сжатии. Данный блок будет реализован с использованием C++ без применения сторонних библиотек из-за высоких требований к производительности для любых операций над самим изображением.

Блок спектрального преобразования реализует основной этап работы над исходным гиперспектральным изображением. За основу данного этапа взято целочисленное вейвлет-преобразование. Преимущества целочисленного вейвлет-преобразования:

– низкая вычислительная сложность;

– обеспечивает заметное улучшение производительности по сравнению с другими видами преобразования;

Используемое преобразование так же известно как CDF 5/3, и используется в стандарте JPEG2000, но в отличии от JPEG2000, в котором количество уровней декомпозиции является изменяемым значением, в используемом стандарте их количество фиксировано и равно пяти.

IWT – это обратимое преобразование, которое точно восстанавливает исходное входное изображение при инвертировании (при условии, что преобразованное изображение не страдает от потерь информации).

Блок уменьшения битовой глубины реализует этап понижения битовой частоты, применяемый к изображению после этапа спектрального преобразования. Целью данного этапа является уменьшение глубины бит преобразованного изображения в соответствии с глубиной бит, поддерживаемой двухмерными кодировщиками промежуточных изображений. Сжатие без потерь обычно невозможно, если на данном этапе удаляется любой младший бит.

Блок двумерного кодирования промежуточных изображений - это модуль, получающий на вход изображение пониженной битовой глубины, и создаёт сжатые изображения, используя несколько экземпляров двухмерного кодера. Каждый двухмерный кодер применяет двухмерное дискретное вейвлет-преобразование к каналу изображения с пониженной частотой. Кодированные сегменты расположены в группах, и для каждой группы существует заголовок переменной длины, который используется для описания спектрального преобразования. Группа и связанный с ней заголовок называется коллекцией. Множество коллекций будет передано в блок кодирования конечного изображения

Блок кодирования конечного изображения получает на вход множество коллекций, после чего объединяет их в последовательность которая и будет выходным сжатым изображением.