**2 СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ**

Изучив теоретические аспекты разрабатываемого средства и выработав список требований необходимых для разработки, разбиваем программу на функциональные блоки (модули).

В разрабатываемом приложении можно выделить следующие блоки:

* блок пользовательского интерфейса;
* блок разделения видеозаписей на кадры;
* блок обработки изображения;
* блок операции детектирования;
* блок стандартизации изображений;
* блок классификации;
* блок приведения полученных данных к одному формату;
* блок экспортирования результатов в базу данных.

Структурная схема, иллюстрирующая перечисленные блоки и связи между ними приведена на чертеже ГУИР.400201.067 C1.

Рассмотрим функциональные блоки приложения.

**2.1 Блок пользовательского интерфейса**

Блок пользовательского интерфейса представляет собой совокупность средств, при помощи которых пользователь взаимодействует с приложением.

Для построения интерфейса используется технология Windows Forms - интерфейс программирования приложений, отвечающий за графический интерфейс и представление данных, получение и обработку ввода пользователя и представляет собой событийно-ориентированное приложение, поддерживаемое Microsoft .NET Framework. Здесь пользователь взаимодействует с программой с помощью различных элементов управления. Так как обработка исходной информации происходит в несколько этапов, то интерфейс изменяется по мере прохождения определенных стадий.

**2.2 Блок разделения на кадры**

Блок разделения на кадрывыполняет задачу кадрирования полученных видеозаписей для последующей работы с отдельными изображениями. Он является оберткой ffmpeg – набора программ с открытым исходным кодом, которые позволяют обрабатывать, конвертировать и передавать цифровые аудио- и видеозаписи в различных форматах. В данной программе из набора используется только одноименный модуль ffmpeg – утилита командной строки для конвертирования видеофайла из одного формата в другой. Он вызывается как отдельный процесс, в который передается набор аргументов (путь к папке с видеозаписями для обработки, путь к папке для сохранения результатов, частота преобразования). В результате получается набор папок с названиями соответствующими исходным видео в заданной директории.

**2.3 Блок обработки изображения**

Блок обработки изображения необходим для улучшения количественных и качественных характеристик модулей детектирования и классификации. Здесь происходит преобразования изображений в наиболее удобные для работы форматы и цветовые форматы. Функционал основан на использовании библиотеки EmguCV.

**2.4 Блок детектирования**

Блок детектированияработает на основе алгоритма каскадов Хаара, используя Виолы-Джонса для нахождения знаков на изображении. В методе Виолы-Джонса основу составляют примитивы Хаара, представляющие собой результаты разбития заданной прямоугольной области на наборы разнотипных прямоугольных подобластей.

Метод использует следующие ключевые концепции:

* простые прямоугольные функции, называемые функциями Хаара;
* метод машинного обучения AdaBoost;
* интегральное изображение для упрощения поиска;
* каскадный классификатор для эффективного совмещения множественных функций.

Каскады Хаара – наборы масок, прямоугольных окошек, каждое из которых представляет собой изображение с неким черно-белым узором (комбинацией черных и белых частей). Таких масок может быть неограниченное множество, сложность узоров может также разниться (см. рис. 2.1).

AdaBoost – это алгоритм усиления классификаторов. Усилением классификаторов называют ансамблевый алгоритм обучения, который использует множество алгоритмов обучения, например, деревья решений, и объединяет их. Целью является взять набор или группу слабых классификаторов и объединить их в один сильный.

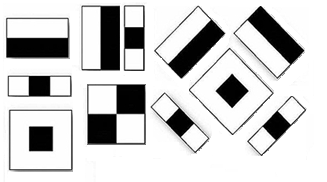


Рисунок 2.1 – Примитивы Хаара [11]

Интегральное представление изображения – это матрица, размерность которой совпадает с размерностью исходного изображения. Элементы этой матрицы рассчитываются по следующей формуле:

II(x, y) = Sum(I(i,j)), (2.1)

где I(i,j) – яркость пикселя исходного изображения.

Из формулы видно, что каждый элемент интегрального изображения II[x, y] содержит в себе сумму пикселей изображения в прямоугольнике от (0,0) до (x, y). Расчет интегрального изображения занимает линейное время, пропорциональное числу пикселей исходного изображения.

В данном блоке используется класс *CascadeClassifier* библиотеки EmguCV. Для работы алгоритма необходимы файлы обученных каскадов. Для обучения необходимо не менее 1000 положительных и столько же отрицательных примеров и около суток для обучения. Результат обучения хранится в xml файле, который и используется данным блоком.

**2.5. Блок стандартизации изображений**

Блок стандартизации изображений необходим для преобразования изображений, полученных на этапе детектирования, к одному формату. Это необходимо для проведения последующей операции классификации. Необходимо привести изображения к одному размеру, выровнять яркость.

**2.6 Блок классификации**

Блок классификациипредставляет собой сверточную нейронную сеть (СНС), которая после обучения на подготовленной выборке производит классификации дорожных знаков на найденных на предыдущем этапе прямоугольных фрагментах изображения. СНС предназначены для эффективного распознавания образов на основе заранее подготовленной базы. Здесь СНС на основе обучающей выборки при подаче изображения будет определять, с какой вероятностью знак на изображении принадлежит к тому или иному классу и на основе этих значений будет выбираться наибольшее, что, скорее всего, означает принадлежность к этому классу. Каждая СНС состоит из нескольких слоев. Среди них могут быть:

1. Входной. На данный слой поступает каждый пиксель исходного изображения. Количество нейронов на данном слое должно быть равно количеству пикселей на классифицируемом изображении. Данный слой выполняет только одну задачу – распределение входных сигналов остальным нейронам. Нейроны этого слоя не производят никаких вычислений;
2. Свертки. Этот слой является основным и обязательным для данного типа нейронной сети. Слой свертки включает в себя для каждого канала свой фильтр, ядро свертки которого обрабатывает предыдущий слой по фрагментам (суммируя результаты матричного произведения для каждого фрагмента). Весовые коэффициенты ядра свертки неизвестны и устанавливаются в процессе обучения;
3. Слой активации. Слой активации обычно логически объединяют со слоем свертки. Результат каждой свертки в скалярном виде попадает на функцию активации в данном слое. Функция активации же представляет собой некую нелинейную функцию. Функция нелинейности в свою очередь может быть любой, традиционно для этого используется функция типа гиперболического тангенса (;
4. Пулинг или слой субдискретизации. Данный слой представляет собой нелинейное уплотнение карты признаков, при этом группа пикселей (обычно размера 2×2) уплотняется до одного пикселя, проходя нелинейное преобразование. Чаще всего для этого используется функция максимума. Операция пулинга позволяет существенно уменьшить пространственный объем изображения. Смысл операции пулинга можно описать так: если на предыдущей операции свертки уже были выявлены некоторые признаки, то для дальнейшей обработки настолько подробное изображение уже не нужно, и оно уплотняется до менее подробного;
5. Выходной. Количество нейронов в этом слое, как правило, равно количеству определяемых классов.  При этом устанавливается соответствие между выходом нейронной сети и классом, который он представляет. Когда сети предъявляется некий образ, на одном из ее выходов должен появиться признак того, что образ принадлежит этому классу. В тоже время на выходах остальных нейронов должны быть признаки того, что образ данному классу не принадлежит.

К преимуществам данного типа нейронной сети можно отнести:

* лучшие показатели в распознавании и классификации изображений;
* гораздо меньшее количество настраиваемых весов (по сравнению с полносвязной нейронной сетью). Используется одно ядро весов для всего изображения. Это приводит к обобщению демонстрируемой информации, а не попиксельному запоминанию каждой показанной картинки, что улучшает качество распознавания;
* удобное распараллеливание вычислений, а, следовательно, возможность использования графических процессоров для ускорения работы;
* относительная устойчивость к повороту и сдвигу распознаваемого изображения;
* обучение при помощи классического метода обратного распространения ошибки.

Недостатком же является большое количество варьируемых параметров сети. Что приводит к сложности конфигурации сети для определенной поставленной задачи. Существует несколько выверенных и прекрасно работающих конфигураций сетей, но не хватает рекомендаций, по которым нужно строить сеть для новой задачи.

**2.7 Блок получения координат**

Блок получения координатсодержит необходимые методы получения географических координат дорожных знаков, используя исходные видеозаписи. В исходных видеозаписях каждый кадр имеет соответствующую запись о координатах транспортного средства в момент записи. Эти данные хранятся в субтитрах и, проведя соответствующее преобразование, используя их можно определить координаты съемки определенного кадра. Используя утилиту ffmpeg и необходимые аргументы можно получить субтитры в текстовом формате txt, в котором каждая запись содержи текущее время записи, географические координаты и время, когда данных дорожный знак был обнаружен на видеозаписи.

**2.8 Блок приведения полученных данных к одному формату**

Блок приведения полученных данных к одному форматунеобходим для стандартизации данных, полученных при обработке видеозаписей в ходе работы программы, таких как название обнаруженного дорожного знака, его географических координат и времени его обнаружения к формату, который пригоден для экспортирования в базу данных. Каждый подготовленный элемент содержит название знака, его координаты и время создания видеозаписи, обработка которого была произведена, что позволит контролировать изменение дорожных знаков на определенных участках дорог.

**2.9 Блок экспортирования результатовв базу данных**

Блок экспортирования результатовв базу данныхсодержит программный код для сохранения полученных результатов в реляционную sql базу данных. Пользователь указывает место хранения данных результатов предыдущего блока. Для сохранения результатов работы программы будет использоваться СУБД Oracle. При попытке записи в базу данных дорожного знака, который был обнаружен повторно, будет произведено обновление времени его обнаружения без полной перезаписи.