РАЗРАБОТКА НЕЙРОННОЙ СЕТИ И АЛГОРИТМА ОПРЕДЕЛЕНИЯ РАССТОЯНИ ДО ОБЪЕКТОВ ДЛЯ УСТАНОВКИ В БЛОК ПОЛЕЗНОЙ НАГРУЗКИ БЕЗЭКИПАЖНОГО КАТЕРА ПРОМЕЖУТОЧНОГО КЛАССА

М.И. Панкратов, А.К. Насонов
Ю.А. Чурсин
Томский политехнический университет
mip15@tpu.ru

Введение

В сентябре 2017 года научно-исследовательская лаборатория телекоммуникаций, приборостроения и морской геологии Томского политехнического университета начала разработку безэкипажного катера промежуточного класса. В катере подразумевается использование самых современных технологий, в том числе нейронные сети. Для обхода препятствий катером и поиском навигационных буев блоком полезной нагрузки, а также измерение расстояния до объектов, создается система технического зрения на основе нейронно-сетевых алгоритмов, с использованием библиотеки Tensorflow.

Работа с TF и настройка нейросети

Работа с TF строится вокруг построения и выполнения графа вычислений. Граф вычислений — это конструкция, которая описывает то, каким образом будут проводиться вычисления. В классическом императивном программировании мы пишем код, который выполняется построчно. В TF привычный императивный подход к программированию необходим только для каких-то вспомогательных целей. Основа TF — это создание структуры, задающей порядок вычислений. Программы естественным образом структурируются на две части — составление графа вычислений и выполнение вычислений в созданных структурах. Катер будет обеспечивать расстановку и сбор в автоматическом режиме навигационных буев в соответствии с заданием миссии. Возможно управление катером в ручном режиме. Одной из ключевых особенностей катера является возможность использования сменных блоков полезной нагрузки. Таким образом, катер сможет выполнять различные задачи, в зависимости от заданных требований и условий. Научной новизной в разработке обладают программно-аппаратные решения для работы с объектами в акватории.

В TF граф состоит из плейсхолдеров, переменных и операций. Из этих элементов можно собрать граф, в котором будут вычисляться тензоры. Тензоры — многомерные массивы, они служат «топливом» для графа. Тензором может быть как отдельное число, вектор признаков из решаемой задачи или изображение, так и целый батч описаний объектов или массив из изображений. Вместо одного объекта мы можем передать в граф массив объектов и для него будет вычислен массив ответов. Работа TF с тензорами похожа на то, как обрабатывает массивы numpy, в функциях которого можно указать ось массива, относительно которой будет выполняться вычисление

Вычислительные графы выполняются в сессиях. Объект сессии (tf.Session) скрывает в себе контекст выполнения графа — необходимые ресурсы, вспомогательные классы, адресные пространства.

Существует два типа сессий — обычные, которые реализованы в tf.Session и интерактивные (tf.InteractiveSession). Разница между ними в том, что интерактивная сессия больше подходит для выполнения в консоли и сразу определяет себя как сессия по умолчанию. Основной эффект — объект сессии не нужно передавать в функции вычисления как параметр. В примерах далее я буду считать, что в данный момент работает интерактивная сессия, которую мы объявили в первом примере, и когда понадобится обращение к сессии, буду обращаться к объекту sess.

В нашей работе было принято решение использовать эту библиотеку как основную, на её основе была использована упрощенная нейронная сеть и были получены первые результаты для определения различных объектов. Процесс обучения сети занял достаточно большое время, но результатом было определение объектов на изображении для начала самых простых, ручка и линейка. В дальнейшем планируется переобучение модели на определение навигационных буев и возможных препятствий на воде. Ниже приведен результат классификации изображения – рисунок 1.

Рис. 1. Тестирование классификатора нейросети

Как видно из рисунка нейросеть смогла определить необходимые для нас объекты и выделила их на изображении, это хороший результат т.к. объектов на картинке несколько, а значит сеть обучилась достаточно хорошо.

Определение расстояния до объекта по изображению

Мы создали сеть и смогли классифицировать объекты, но в акватории для поиска и сбора навигационных буев этого недостаточно. Так как камера используется с одной двояковыпуклой линзой. Чтобы решить эту задачу, необходимо использовать схему схода лучей в тонкой линзе из геометрической оптики – рисунок 2.

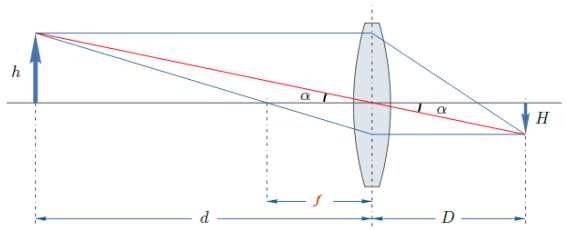

Рис. 2. Схема схода лучшей в тонкой линзе

На этой схеме d — расстояние от линзы до объекта, D — расстояние от линзы до изображения объекта (на матрице или плёнке), а f — фокусное расстояние линзы. Формула тонкой линзы приведена ниже:

$$\frac{1}{d}+\frac{1}{D}=\frac{1}{f}$$

Теперь ещё раз посмотрим на оптическую схему: h — это линейный размер объекта съёмки, а H — размер его уменьшенного изображения. Нетрудно заметить, что $h = d \tan \alpha$, а $H = D \tan \alpha$ (это следует из свойств прямоугольного треугольника). Подставив эти величины в формулу тонкой линзы, увидим, что $\tan \alpha$ сокращается, и в результате получим следующее уравнение:

$$1+\frac{h}{H}=\frac{d}{f}$$

Неудобная величина D ушла, а остальные мы знаем или можем легко вычислить. На основе этого уравнения получаем вот такую формулу расстояния до объекта: $\boldsymbol{d = (f(H + h)) / H}$.

H — это размер изображения классифицированного объекта на матрице камеры. По фотографии мы можем подсчитать его в пикселях, но, лучше использовать конкретные физические размеры, которые имеет матрица камеры raspberry pi. b составляет 0,37 × 0,27 см, а разрешение — 2592 × 1944 пикс. Наша фотография не была кадрирована или повёрнута, поэтому мы можем узнать точный линейный размер изображения дома на матрице, составив пропорцию.

Зная физический размер матрицы и количество пикселей по длинной стороне снимка, делаем такой расчёт: 2592 / 0,37 (размер матрицы в сантиметрах), и получаем правильное разрешение — 7005 пикс./см. Получается, что высота изображения объекта на матрице составляет 0,15 см, или 0,0015 м. Осталось только выяснить фокусное расстояние, но для этого, к счастью, ничего считать не нужно: оно сразу прописывается при съёмке в метаданных фотографии (EXIF) – рисунок 3.

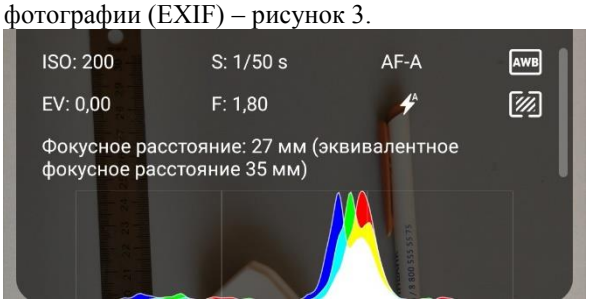

Рис. 3. Метаданные фотографии

Фокусное расстояние при съёмке составляло 27 мм, или 0,027 м. Узнаем размер ручки,, он равен 7 см или 0,07м Теперь у нас есть все данные для расчёта. Подставляем их в формулу $d = (f(H + h)) / H$ и получаем: d = (0,027 (0,0015 +0,07)) / 0,0015 = 1,2 м.

Исходя из расчетов погрешность относительно небольшая, т.к. съемка производилась с расстояния 1.3м. такой результат нас устраивает, потому что на больших расстояниях погрешность будет уменьшаться.

Заключение

На данный момент решается задача переобучения нейросети на определение реальных объектов и измерения расстояния до них, а также дообучение классификатора, чтобы увеличить точность определения объектов. Также разрабатывается алгоритм взаимодействия между нейросетью и управлением блоком полезной нагрузки для сбора и расстановки буев.

Список использованных источников

1. Барский А.Б. Нейронные сети. Распознавание, управление, принятие решений// - Москва: Финансы и статистика, 2004. – С.34-50.
2. Li Deng, Dong YuDeep Learning: Methods and Applications// Foundations and TrendsR in Signal Processing Vol. 7, Nos. 3–4 (2013) 197–387.
3. Deep Learning Papers Reading Roadmap [Электронный ресурс] режим доступа: https://github.com/floodsung/Deep-Learning-Papers-Reading-Roadmap