DOI: 10.17117/na.2016.09.01.393

Поступила (Received): 03.09.2016

<http://ucom.ru/doc/na.2016.09.01.393.pdf>

Гольдштейн М.А.
Применение верификаторов программного
обеспечения для анализа перемещений
объектов на видеоизображениях

Goldshtein M.A.
Verificators application for analyzing objects' movements in video

Современная востребованность анализировать видеоконтент ставит задачу создания автоматических средств, способных самостоятельно и быстро анализировать большие объемы информации. В статье рассмотрена идея использования верификаторов программного обеспечения в системах компьютерного зрения и предложен алгоритм анализа перемещения детектированных на видео объектов. Координаты объектов, полученные с помощью алгоритмов компьютерного зрения передаются в верификатор, который производит проверку выполнения детектированными объектами заданных правил

Ключевые слова: компьютерное зрение, верификация, проверка программных моделей, детектирование

Modern demand for video content analysis puts the task of creation of automatic tools that can analyze large amount of information on their own and quickly. The article considers the idea of using the verifier software systems, computer vision, and an algorithm for the analysis of movement of the detected object on the video. The coordinates of the objects obtained by computer vision algorithms are transmitted to the verifier, who verifies the detected objects performing predetermined rules

Key words: computer vision, verification, verification of software models, detection

Гольдштейн Михаил Анатольевич

Студент

Московский государственный технический университет им. Н.Э. Баумана
г. Москва, 2-я Бауманская ул., 5

Goldshtein Michael Anatolievich

Student

Moscow state technical university named N.E. Bauman
Moscow, Baumanskaya 2-ya st., 5

Введение. Компьютерное зрение плотно вошло в нашу жизнь в самых различных отраслях. Видеосъемка плотно вошла в жизнь современного человека. Камеры наблюдения на улицах и видеорегистраторы в автомобилях, камеры в смартфонах и компьютерах генерируют огромное количество видеоконтента. Скорость появления новой информации сделала невозможным анализ содержимого видеосъемки “вручную”, операторами [1, с. 71]. Появилась необходимость средств автоматического анализа видеоконтента. Технологии компьютерного зрения позволяют извлекать полезную информацию из видеоизображений. Поиск оптимального способа анализа полученной информации является актуальной проблемой. Цель данной работы – решение задачи отслеживания объектов

на видео, а также проверка эффективности и удобства использования инструментов для автоматической проверки моделей для анализа перемещений детектированных объектов.

Алгоритм отслеживания. Детектированием объектов на видеоизображениях называется поиск положений объектов определенного класса в кадре. Положением объекта называют набор координат пикселей, соответствующих объекту либо координаты окружающих объект прямоугольников. В нашей работе требуется получать координаты окаймляющих прямоугольников. Алгоритм детектирования людей основан на использовании гистограмм ориентированных градиентов (HOG-признаков). Данный метод использует описание формы объектов за счет распределения градиентов интенсивности пикселей, относящихся к объекту, не учитывая значения градиента в конкретной точке. Градиентом называется аппроксимация градиентной функции яркости, значение которой известно только в узлах сетки – в пикселях, относящихся к заданной точке некоторой заданной схемы.

Минимальным элементом HOG-дескриптора называется блок (block) – область изображения, определенной формы. Блок делится на ячейки (cells), которые состоят из пикселей. Ячейкам сопоставляются гистограммы ориентаций градиентов (углы наклона от горизонтали), с определенным количеством полос (bins), направления наклонов считаются «беззнаковыми».

В библиотеке компьютерного зрения с открытым исходным кодом OpenCV имеется реализация HOG-детектора, использующая в качестве классификатора линейную машину опорных векторов. Линейный SVM-классификатор является вектором коэффициентов, полученных из уравнения гиперплоскости, разделяющей пространство признаков. Для осуществления детектирования необходимо задать используемую модель классификатора. Для этого используется метод setSVMDetector.

Рис. 1. Результат работы HOG-детектора

SVMdetector – матрица, образованная коэффициентами разделяющей гиперплоскости. Размер вектора должен на единицу превышать длину HOG-дескриптора детектируемого окна, либо быть равным ей. Библиотека OpenCV содержит линейные SVM-модели, для детектирования пешеходов, обученные на базах INRIA и Daimler, которые можно получить с помощью методов getDefaultPeopleDetector и getDaimlerPeopleDetector. Непосредственно детектирование производится с помощью метода detect.

Детектированные объекты необходимо отслеживать. Определение местоположения движущихся объектов на последовательности кадров называется трекингом. Для дальнейшего анализа перемещений объектов, будем определять их координаты, записывать последовательность найденных координат в файл, параллельно с этим в отдельном окне будут выводиться треки, в виде точек, соответствующих каждому детектированию объекта. Каждому объекту присваивается свой цвет точек.

Рис. 2. Визуализация трекинга

Проверка моделей. Проверка моделей или model checking – это алгоритм верификации программ, исполняемых компьютером и контроль их корректности. Проверка производится путем ввода в специальную программу – верификатор, описания модели исследуемой системы и спецификации требований. Верификатор производит проверку соответствия поведения модели заданным требованиям и в случае обнаружения ошибки представляет контрпример, или сценарий, ведущий к нежелательному поведению системы. После этого можно исправить модель и спецификацию и продолжить проверку. Верификация заканчивается, когда ни одной ошибки не найдено. По мере роста сложности и размеров системы модель и спецификация уточняются и дополняются.

Алгоритм проверки моделей заключается в последовательном переборе всех возможных состояний исследуемой модели и анализе выполнения требуе-

мых свойств в каждом из этих состояний. Данный метод также называют анализом достижимости, он позволяет обнаружить возможность попадания системы в запрещенные состояния и избежать возникновения подобных ситуаций.

Инструменты для проверки моделей. На сегодняшний день существует огромное количество инструментов, позволяющих проводить проверку моделей, они отличаются способом моделирования, используемой логикой, исходным предназначением. Рассмотрим основные известные верификаторы.

Spin. Один из первых инструментов формальной верификации распределенных систем. Разрабатывался в Bell Labs, начиная с 1980г. В настоящее время распространяется свободно. SPIN – это широко распространенный инструмент с открытым исходным кодом, используемый по всему миру. В SPIN имеется три режима работы: режим симуляции, осуществляющий прототипирование со случайным моделированием; режим верификации, в котором доказываемая корректность модели; режим аппроксимации, способный подтвердить правильность протоколов [3, с. 586]. SPIN использует собственный язык PROMELA (PROcess MEta LAnguage), с помощью него можно специфицировать описание исследуемой программы. PROMELA поддерживает дискретные типы данных и функции для работы с многопоточностью. SPIN можно использовать для отслеживания логических ошибок в операционных системах, распределенных и параллельных алгоритмах, а также для проверки различных протоколов. Свойства для SPIN описываются с помощью линейной темпоральной логики (LTL), часть спецификации модели может быть выражена на языке С. Это позволяет верифицировать высокоуровневые темпоральные свойства. Для работы с системой SPIN обычно используют графическую оболочку Ispin. Первым шагом работы является создание высокоуровневой модели алгоритма. После исправления ошибок проводят эмуляцию, чтобы удостовериться в корректном поведении модели. Далее SPIN используется, чтобы сгенерировать программу проверки спецификации высокого уровня. Полученную программу необходимо скомпилировать. При обнаружении контрпримеров, они возвращаются в эмулятор, чтобы можно было установить и устранить их причину. На основе SPIN создано большое количество верификаторов, используемых для проверки доменов. Данная система предназначена для решения крупномасштабных задач.

NuSMV. New Symbolic Model checker. Символьный верификатор моделей, разработан компаниями Formal Methods Group ITC-IRST, the Model Checking group Carnegie Mellon University, the Mechanized Reasoning Group в University of Genova и Mechanized Reasoning Group в University of Trento. NuSMV – это обновленная версия SMV, он разработан как система с открытой архитектурой, предназначенная для верификации моделей, которую можно использовать для проверки промышленных разработок, как ядро сложных верифицирующих инструментов и как основа для тестирования других технологий формальной верификации. NuSMV производит верификацию, комбинируя техники использования BDD и проверку моделей, основанную на SAT (SAT-based model checking). Проверяет требования корректности, выраженные в ветвящейся темпоральной логике CTL. В отличие от SMV обеспечивает возможности взаимодействия конечных автоматов, реализации методов декомпозиции, анализа инвариантов [4, с.

43]. Программы в NuSMV представляются набором модулей, которые описывают свое внутреннее состояние и систему переходов, в зависимости от входных воздействий.

Visual Prolog. Visual Prolog—это объектно-ориентированное расширение языка программирования PDC Prolog, развивавшегося из Turbo Prolog (Borland), семейства Prolog, а также система визуального программирования датской фирмы Prolog Development Center. Visual Prolog позволяет автоматизировать построение сложных процедур и освободить программиста от выполнения тривиальных операций. В системе Visual Prolog проектирование пользовательского интерфейса и связанных с ним окон производится в графической среде. С созданными объектами могут работать различные Кодовые Эксперты (Code Experts), использующиеся для генерации базового и расширенного кодов на языке Prolog, необходимых для обеспечения их функционирования. Мощность языка Prolog в сочетании с системой пользовательских интерфейсов упрощает разработку систем, основанных на знаниях, систем поддержки принятия решений, планирующих программ, развитых систем управления базами данных и т. д.

Анализ перемещений объектов. Среди рассмотренных систем наиболее подходящей для анализа перемещений объектов на видео оказалась система Visual Prolog. Остальные верификаторы заточены на анализ распределенных систем с очень большим количеством состояний, что не является необходимым и удобным свойством для системы анализа видеоизображений. В итоговом варианте системы анализа перемещений объектов Visual Prolog принимает множество последовательных координат детектированных объектов и сохраняет их в списки, соответствующие каждому исследуемому объекту. В другой список заносится набор координат ключевых точек, порядок прохождения которых необходимо проверить. Далее Visual Prolog автоматически анализирует прохождение ключевых точек исследуемым объектом и выносит решение о выполнении объектом исследуемого правила.

Заключение. В работе рассмотрена возможность применения инструментов верификации в системах компьютерного зрения. Были рассмотрены несколько популярных средств верификации и предложен алгоритм анализа перемещений детектированных на видео объектов. В дальнейшей работе будут разработаны алгоритмы, позволяющие автоматически анализировать перемещения нескольких объектов и проверять выполнение сложных правил.

Список используемых источников:

1. Девятков В.В., Алфимцев А.Н. Нечёткая конечно-автоматная модель интеллектуального мультимодального интерфейса // Проблемы управления. 2011. № 2. С. 69-77.
2. Алфимцев А.Н., Хаев Ф.И. Гендерные различия в восприятии информации и организации пользовательского интерфейса компьютерных систем // Автоматизация. Современные технологии. 2015. № 6. С. 25–28.
3. Marc Frappier, Benoit Fraikin, Romain Chossart. Comparison of Model Checking Tools for Information Systems // Formal Methods and Software Engineering. ICFEM, 2010. P. 581-596.
4. Marek Tomastik. Comparison of parallel LTL model checkers // Bachelor thesis. Masaryk University. Brno. 2013. 63 p.

5. Вельдер С.Э., Лукин М.А., Шалыто А.А., Яминов Б.Р. Верификация автоматных программ. 2011. 246 с.
6. Кулямин В.В. Методы верификации программного обеспечения // Информационно-телекоммуникационные системы. 2008. 117 с.

© 2016, Гольдштейн М.А.

Применение верификаторов программного
обеспечения для анализа перемещений объектов на
видеоизображениях

© 2016, Goldshtein M.A.

Verificators application for analyzing objects'
movements in video