Информационные процессы, Том 18, № 4, 2018, стр. 366–375
© 2018 Смагина, Шепелев, Ершов.

МАТЕМАТИЧЕСКИЕ МОДЕЛИ, ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ

Об оценке качества оптического потока для обнаружения препятствий [1]

А.А. Смагина, Д.А. Шепелев и Е.И. Ершов

Институт проблем передачи информации, Российская академия наук, Москва, Россия

Поступила в редколлегию 20.11.2018

Аннотация—Рассмотрен вопрос выбора алгоритма вычисления плотного оптического потока (ОП) для решения задачи обнаружения препятствий в режиме реального времени. Для оценки качества алгоритмов мы использовали метрику KITTI, а также предложили её модификации для учета распределения ошибок вычисления по области определения изображения ОП. Вычислительные эксперименты показали, что упорядочение алгоритмов по этим метрикам отличается, что может быть существенным в контексте задачи обнаружения препятствий. В результате получен набор эксплуатационных характеристик алгоритмов вычисления ОП.

КЛЮЧЕВЫЕ СЛОВА: оптический поток, обнаружение препятствий, уход от столкновений, KITTI

При автоматическом пилотировании транспортных средств приходится решать задачу обнаружения препятствий на дороге, что, в свою очередь, требует оценки расстояния до объектов на пути движения. Один из типичных подходов к этому – использование алгоритмов плотного оптического потока. В настоящее время создано множество алгоритмов вычисления оптического потока (по запросу «optical flow algorithm» ресурс google.scholar.com выдаёт более двух с половиной миллионов результатов) и перед разработчиком стоит задача выбора наиболее подходящего. Сравнение точности различных алгоритмов можно найти, например, на интернет-ресурсах KITTI [1] и Middlebury [2]. При этом в качестве метрик ранжирования берутся универсальные, не специфичные для конкретных применений: доля «плохих» (превышающих абсолютный или относительный пороги отклонений) пикселей, сумма модулей отклонений и корень из суммы квадратов отклонений по всем пикселям.

В то же время существует гипотеза, что для задач обнаружения дорожных препятствий более корректным будет применение специальных проблемно-ориентированных метрик, например предложенной в работе Сидорчука и соавторов [3]. Эта метрика позволяет учесть важность правильного определения расстояния до препятствий именно в коридоре движения транспортного средства, назначая каждому пикселю изображения некоторый аналитически задаваемый вес, спадающий по мере удаления от камеры.

Существует и другой подход, при котором качество алгоритма измеряется не напрямую, а в контексте работы всей системы обнаружения препятствий [4]. Этот подход позволяет оценить влияние качества работы оптического потока на целевое качество системы, однако на этапе проектирования этот способ оценки недоступен.

Сравнение алгоритмов по времени работы также сопряжено с рядом технических трудностей, часть которых устранить технически трудно. Во-первых, зачастую время выполнения указывается для входных изображений разного размера. Для решения этой проблемы в работе [5] предложен способ сопоставления алгоритмов по числу оценок значения оптического

[1] Работа выполнена при финансовой поддержке гранта РНФ № 14-50-00150.

потока на изображении заданного размера за единицу времени. Во-вторых, измерения проводятся на вычислителях различной производительности и архитектуры. Проблему различия производительности можно частично устранить путём нормировки на мощность вычислителя. И, наконец, в-третьих, результаты измерения в значительной степени зависят от качества реализации и компилятора, что и вовсе является трудно устранимым человеческим фактором.

В настоящей работе проведено сравнение алгоритмов, рассчитанных на работу на единственном ЦПУ без графического процессора. В качестве тестовых данных мы использовали общедоступный набор данных KITTI [1] с незначительными модификациями, которые детальнее будут описаны ниже. В рамках исследования был найден исходный код всех вышеперечисленных алгоритмов, что оказалось важно для сопоставления их производительности. Для оценки качества ОП мы использовали метрику KITTI и её модификации для учёта структуры ошибки.

1. МЕТРИКИ ОЦЕНКИ КАЧЕСТВА ОП

Согласно работе [1], качество ОП вычисляется как доля «плохих» пикселей ОП, назовем это «метрика KITTI». Плохим пикселем считается тот, вектор значений которого отличается от правильного на величину больше 3 пикселей или больше чем на 5%. Как видно, данная метрика не чувствительна к расположению плохих пикселей в области расположения, что может оказаться критичным для задачи обнаружения препятствий.

В работе [3] предложен новый метод оценки качества ОП в зависимости от распределения ошибок и удалённости от камеры. В предположении, что большую часть времени ТС движется прямолинейно и столкновение возможно с объектами, находящимися прямо перед ним, было предложено уменьшать влияние ошибки по мере удаления от центра изображения (для летательных аппаратов) или от вертикальной оси изображения (для наземных ТС). Правило изменения влияния при удалении от центральной оси имеет смысл выбирать нелинейной, так, чтобы вес начинал особенно быстро спадать за границами полосы автодороги, по которой движется ТС. В нашей работе удаление от центральной оси задаётся углом отклонения луча из фокуса камеры, проходящего через пиксель, от оптической оси камеры, а вес пикселя спадает как косинус, возведенный в некоторую степень, являющуюся параметром.

Пусть карта глубины ОП z_a, построенная алгоритмом, и истинная карта z_t. Тогда функция $I(i,j)$ значимости каждого пикселя (i,j) вычисляется следующим образом:

$$I(i,j) = \frac{\cos^k \alpha}{\min(z_t(i,j), z_a(i,j))}, \tag{1}$$

где $\alpha = \alpha(i,j)$ – угол отклонения пикселя (i,j) от оптической оси камеры (центральной оси) по горизонтали, а k – действительное число, позволяющее регулировать “крутизну” линий уровня в плоскости движения беспилотного наземного транспортного средства (смотри рисунок 1). В нашей работе мы выбрали $k = 15$.

Метрика (1) для оценки качества алгоритмов ОП предполагает, что задача восстановления глубины решена. Тем не менее эта задача, в отличии от случая стереопары, является боле сложной, поскольку неизвестно каким образом переместилась система координат камеры между первым и вторым снимком. В своей работе для оценки глубины мы использовали программное решение COLMAP [6]. Анализ карты глубины позволяет оценивать качество в метрической системе, что упрощает анализ результатов, подробнее смотри [3]. Однако алгоритм вычисления глубины вносит свою ошибку в результирующую карту глубины, поэтому итоговое сравнение производится опосредовано, что надо иметь ввиду при реализации такого подхода.

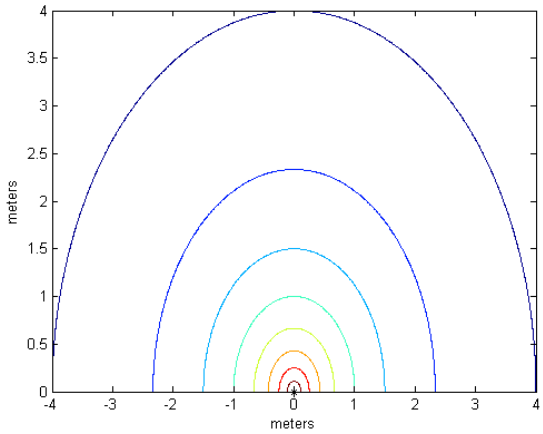


Рис. 1. Линии уровня при $k = 1$

Существуют и другие способы учесть структуру ошибки. Предметом исследования является проверка изменится ли упорядочение алгоритмов по качеству, если учитывать то каким образом распределена ошибка. Мы провели эксперимент с объединенной метрикой, в которой определяется доля «плохих» пикселей (как в метрике KITTI), но при этом вносимый этими пикселями вклад в ошибку определяется положением пикселя на изображении, как в 1:

$$K^f = \frac{\sum_{i,j \in z_t} E(i,j) \cdot \cos^k \alpha(i,j)}{\sum_{i,j \in z_t : z_t(i,j) \neq 0} \cos^k \alpha(i,j)}, \tag{2}$$

где $E(i,j)$ – карта «плохих» пикселей, а условие $z_t(i,j) \neq 0$ является проверкой наличия данных о глубине в данном пикселе. Будем называть эту метрику «KITTI+цилиндр»

Другим аспектом проверки является не расположение плохого пикселя, а концентрация их в некоторой окрестности. В случае если ошибка вычисления ОП сконцентрирована в некоторой плохо области, то, как было сказано ранее, это нежелательное поведение может в дальнейшем привести к существенным ошибкам детекции препятствий. Для проверки выраженности этого эффекта у рассматриваемых алгоритмов мы выполняем морфологическую операцию закрытия над $E(i,j)$ с квадратным окном стороной 4 (это позволяет объединить близлежащие плохие пиксели и компенсировать разреженность истинных данных, полученных с лидара), после – открытие с квадратным окном стороной 16 (для устранения малых областей плохих пикселей), а только затем выполняем подсчет плохих пикселей. Параметры морфологической фильтрации выбраны с учетом разреженной структуры карты глубины, полученной с лидаров и структуры ошибки рассматриваемых алгоритмов оптического потока. Идея заключается в том, что после такой операции большие связанные кластеры плохих пикселей не изменится, а небольшие кластеры будут удалены. Назовём такую метрику «KITTI+морфология».

2. НАБОР ДАННЫХ И РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

При выборе алгоритмов для исследования мы руководствовались следующими правилами отбора: 1) реализация алгоритма должна быть в открытом доступе; 2) алгоритм реализован на языках C/C++/python; 3) реализация использует только CPU; 4) заявленное время работы реализации меньше минуты (что приемлемо с учетом потенциальных возможностей ускорения вычислений).

Вся необходимая информация была получена из таблицы KITTI [1] и затем проверена для каждого алгоритма. Из 107 реализаций алгоритмов под вышеприведенные требования подо-

шли всего лишь 8: PolyExpand (реализация из opencv), CPM-Flow [7], HAOF [11], H+S_ROB [13], Deepflow [10], Epicflow [9], TVL1_ROB [12] и PPM [8].

Для сравнения алгоритмов мы вынужденно использовали обучающую выборку из набора KITTI (flow 2015), так как тестовая находится в закрытом доступе. Истинные значения для ОП получены с помощью лидара, установленного на крыше автомобиля. Вертикальный угол обзора лидара значительно меньше чем у камеры, поэтому на эталонных картах глубины присутствуют только точки, которые соответствуют объектам, находящимся ниже горизонтальной плоскости, проходящей через лидар. В результате истинное значение глубины представляется разреженными неравномерно распределенными точками.

Форма записи ОП, который используется в KITTI, представляет собой трехканальное изображение, в каждом пикселе которого в первом и во втором каналах хранятся значения компонент ОП, а в третьем канале валидность для данного пикселя.

В KITTI представлено два вида достоверных оптических потоков: с включенными и исключенными значениями ОП для пикселей, входящих в зоны окклюзии. Кроме того, в рейтинге приводятся упорядочения алгоритмов не только для вычисленного ОП, но и для ОП с интерполяцией в тех пикселях, где алгоритмы не смогли его вычислить.

По данным из KITTI для каждой метрики можно составить 4 списка упорядочения алгоритмов: 1) без интерполяции, без учета окклюзионных пикселей (EN); 2) без интерполяции, с учетом окклюзионных пикселей (EA); 3) с интерполяцией, с без учета окклюзионных пикселей (IN); 4) с интерполяцией, с учетом окклюзионных пикселей (IA). Оказалось, что таблицы KITTI для случаев EN и EA одинаковы, как таблицы IN и IA.

Алгоритмы запускались на платформе Linux Mint 18.1 Cinnamon 64-bit. Все вычисления были осуществлены на устройстве с четырех ядерным процессором Intel Core i7-4790 3.6 ГГц, с объемом оперативной памяти в 15.6 ГБ.

В работе приведены упорядочения алгоритмов (смотри таблицу 1) по метрикам: 1) KITTI; 2) KITTI+морфология; 3) KITTI+цилиндр, согласно полученным результатам вычисления ОП с помощью этих алгоритмов. Графически результаты оценки качества алгоритмов по вышеуказанным метрикам для четырех разных случаев (EN, EA, IN, IA) представлены на рисунках 2, 3, 4 и 5. Заметим, что результаты алгоритма PPM были предварительно отфильтрованы, так как по невыясненным причинам, он записывал очевидно невалидные значения ОП ($u = -512$ и $v = -512$) как валидные.

Из таблицы 1 видно, что при упорядочении алгоритмов по метрике KITTI+цилиндр полученный порядок относительно метрики KITTI остается неизменным для случаев без интерполяции ОП (EN и EA), а при интерполяции (IN и IA) порядок меняется незначительно (EpicFlow и CPM меняются местами, однако разница между ними не очень велика, см. графики 4, 5).

При упорядочении алгоритмов по метрике KITTI+морфология изменение положения алгоритмов по сравнению с KITTI заметнее. Так, в нижней половине таблицы упорядочения для всех случаев (EN, EA, IN, IA) алгоритм PolyExpand поднимается на 2 позиции относительно ранга в KITTI. Это вызвано тем, что области ошибок были меньше по площади и менее плотными по сравнению с результатами алгоритмов HAOF и TVL1_ROB. А для случаев IN и IA алгоритм PPM поднялся на 2 позиции вверх и, таким образом, в новом рейтинге становится вторым после CPM, который в свою очередь обогнал DeepFlow2 и поднялся на первое место (см. таблицу 1). Аналогично случаю с PolyExpand, распределение ошибок для алгоритмов PPM и CPM оказалось менее плотным и меньшим по площади по сравнению с DeepFlow2 и EpicFlow.

Согласно графикам 2, 3, 4, 5 самыми быстрыми алгоритмами являются PolyExpand и CPM. Таким образом, алгоритм CPM оказывается не только одним из самых быстрых, но и одним из наиболее точных по всем представленым метрикам. А алгоритм PolyExpand, реализация

которого представлена в известной библиотеке OpenCV, оказывается самым быстрым, а в упорядочении по метрике KITTI+морфология для случаев IN и IA занимает 5 место из 8, т.е. оказывается не самым худшим при его скорости работы.

Таблица 1. Ранжирование алгоритмов по различным метрикам качества. В скобках для метрик KITTI+морфология и KITTI+цилиндр приведены сдвиги позиций алгоритма относительно ранжирования по метрике KITTI. Зеленым цветом отмечены алгоритмы, которые в новых метриках улучшили свои позиции по сравнению с ранжированием по метрике KITTI. Аналогично красным отмечены алгоритмы положение которых в новой метрике ухудшилось.

Без интерполяции, без учета окклюзионных пикселей (EN)			Без интерполяции, с учетом окклюзионных пикселей (EA)		
KITTI	KITTI+морфология	KITTI+цилиндр	KITTI	KITTI+морфология	KITTI+цилиндр
PPM	PPM (0)	PPM (0)	PPM	PPM (0)	PPM (0)
CPM	CPM (0)	CPM (0)	CPM	CPM (0)	CPM (0)
DeepFlow2	DeepFlow2 (0)	DeepFlow2 (0)	DeepFlow2	DeepFlow2 (0)	DeepFlow2 (0)
EpicFlow	EpicFlow (0)	EpicFlow (0)	EpicFlow	EpicFlow (0)	EpicFlow (0)
HAOF	PolyExpand (2)	HAOF (0)	HAOF	PolyExpand (2)	HAOF (0)
TVL1_ROB	HAOF (-1)	TVL1_ROB (0)	TVL1_ROB	HAOF (-1)	TVL1_ROB (0)
PolyExpand	TVL1_ROB (-1)	PolyExpand (0)	PolyExpand	TVL1_ROB (-1)	PolyExpand (0)
H+S_ROB	H+S_ROB (0)	H+S_ROB (0)	H+S_ROB	H+S_ROB (0)	H+S_ROB (0)
С интерполяцией, без учета окклюзионных пикселей (IN)			С интерполяцией, с учетом окклюзионных пикселей (IA)		
KITTI	KITTI+морфология	KITTI+цилиндр	KITTI	KITTI+морфология	KITTI+цилиндр
DeepFlow2	CPM (1)	DeepFlow2 (0)	DeepFlow2	CPM (1)	DeepFlow2 (0)
CPM	PPM (2)	EpicFlow (1)	CPM	PPM (2)	EpicFlow (1)
EpicFlow	DeepFlow2 (-2)	CPM (-1)	EpicFlow	DeepFlow2 (-2)	CPM (-1)
PPM	EpicFlow (-1)	PPM (0)	PPM	EpicFlow (-1)	PPM (0)
HAOF	PolyExpand (2)	HAOF (0)	HAOF	PolyExpand (2)	HAOF (0)
TVL1_ROB	HAOF (-1)	TVL1_ROB (0)	TVL1_ROB	HAOF (-1)	TVL1_ROB (0)
PolyExpand	TVL1_ROB (-1)	PolyExpand (0)	PolyExpand	TVL1_ROB (-1)	PolyExpand (0)
H+S_ROB	H+S_ROB (0)	H+S_ROB (0)	H+S_ROB	H+S_ROB (0)	H+S_ROB (0)

Также мы сравнили времена работы алгоритмов с указанными на сайте KITTI (таблица 2). Из таблицы 2 видно, что полученые времена, а также упорядочение алгоритмов по времени заметно отличается от KITTI.

Таблица 2. Ранжирование алгоритмов по среднему времени работы. В скобках указано время работы относительно быстрейшего алгоритма в столбце. Последний столбец – вычислительные мощности, указанные на сайте KITTI.

Метод	Время работы в сек. (время работы относительно самого быстрого алгоритма)	Указанное в KITTI время работы в сек. (время работы относительно самого быстрого алгоритма)	Вычислительные мощности указанные в KITTI
PolyExpand	0.69 (1)	1 (1)	1 core @ 2.5 Ghz (C/C++)
CPM	2.72 (3.94)	4.2 (4.2)	1 core @ 3.5 Ghz (C/C++)
HAOF	5.94 (8.61)	16.2 (16.2)	1 core @ 2.5 Ghz (C/C++)
H+S_ROB	28.73 (41.64)	8 (8)	4 cores @ 2.5 Ghz (C/C++)
DeepFlow2	32.08 (46.49)	17 (17)	1 core @ >3.5 Ghz (Python + C/C++)
EpicFlow	41.68 (60.41)	15 (15)	1 core @ >3.5 Ghz (C/C++)
TVL1_ROB	55.73 (80.77)	3 (3)	4 cores @ 2.5 Ghz (C/C++)
PPM	61.95 (89.78)	17.3 (17.3)	1 core @ 2.5 Ghz (C/C++)

Рис. 2. График средней величины ошибки от среднего времени работы алгоритмов без интерполяции и без учета окклюзионных пикселей. Треугольниками обозначены значения по метрике KITTI, кругами – KITTI-морфологии, квадратами – KITTI-цилиндра.

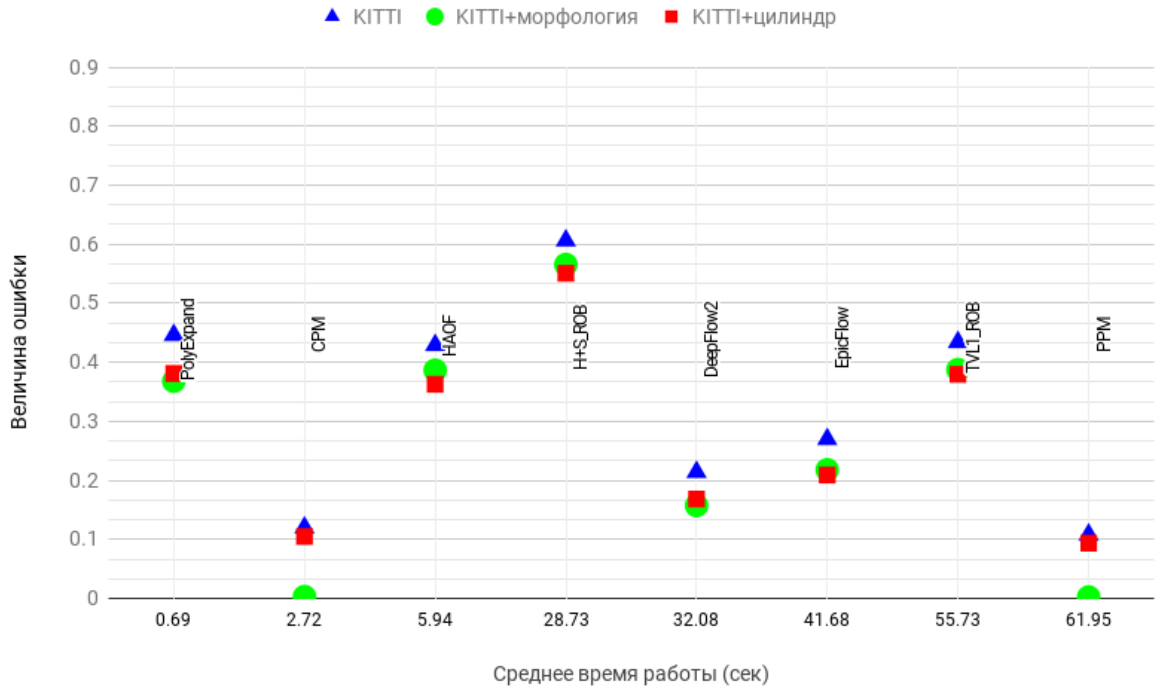


Рис. 3. График средней величины ошибки от среднего времени работы алгоритмов без интерполяции, но с учетом окклюзионных пикселей. Треугольниками обозначены значения по метрике KITTI, кругами – KITTI-морфологии, квадратами – KITTI-цилиндра.

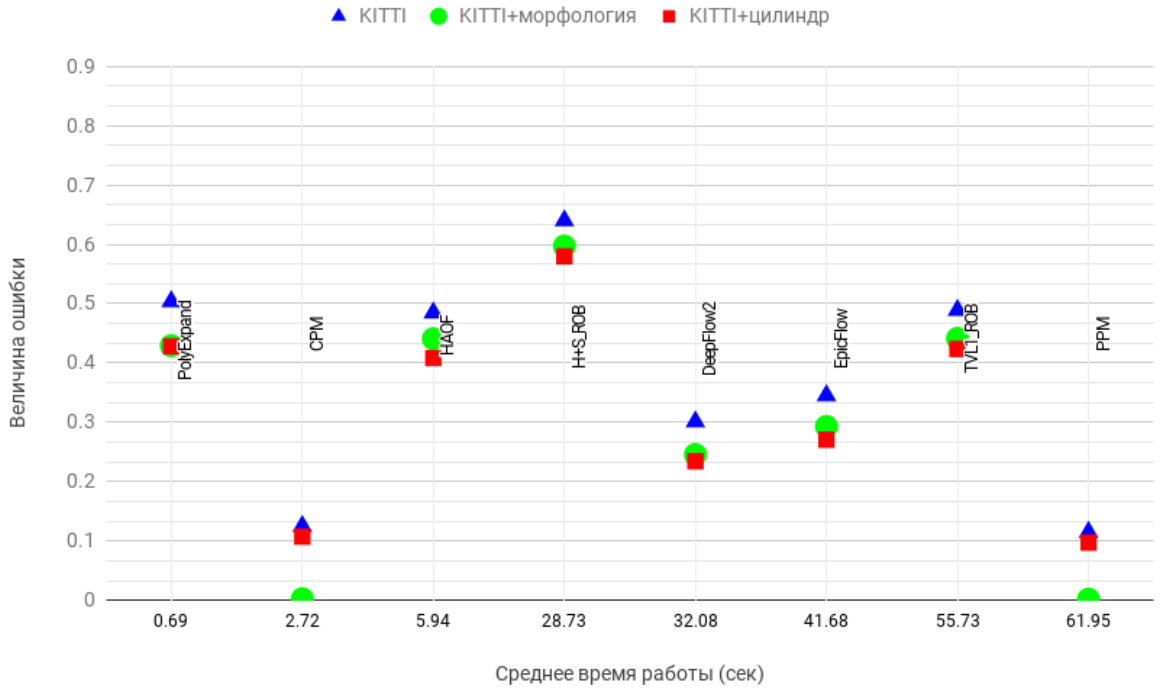


Рис. 4. График средней величины ошибки от среднего времени работы алгоритмов с интерполяцией вычисленных оптических потоков и без учета окклюзионных пикселей. Треугольниками обозначены значения по метрике KITTI, кругами – KITTI-морфологии, квадратами – KITTI-цилиндра.

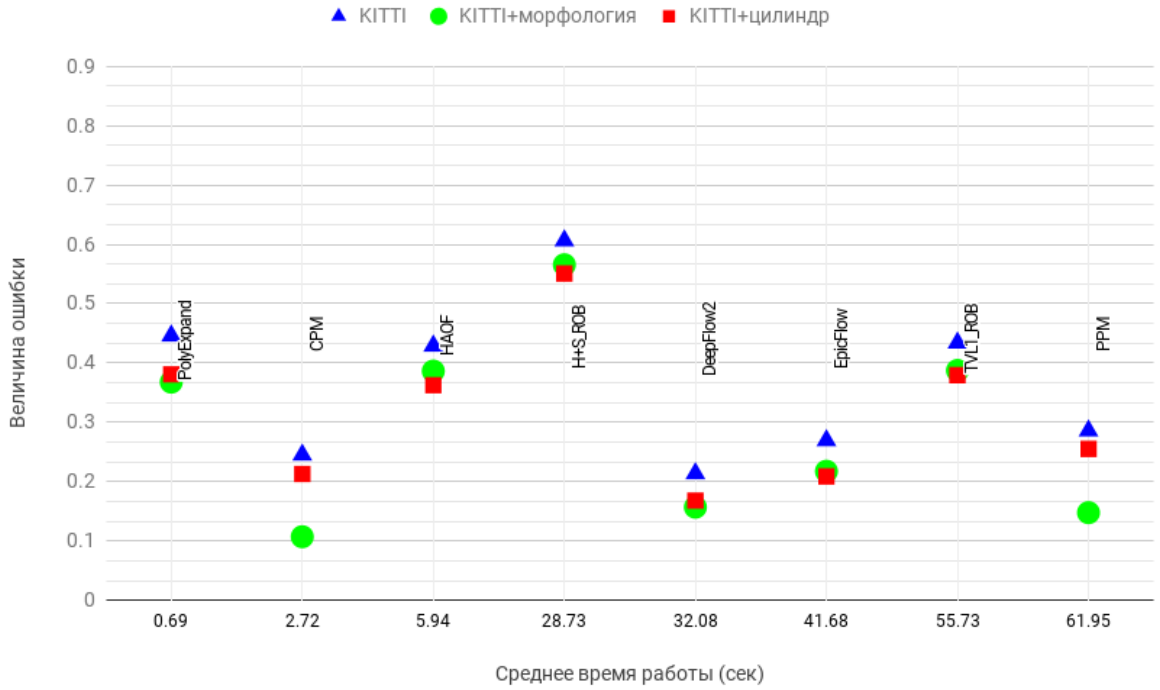
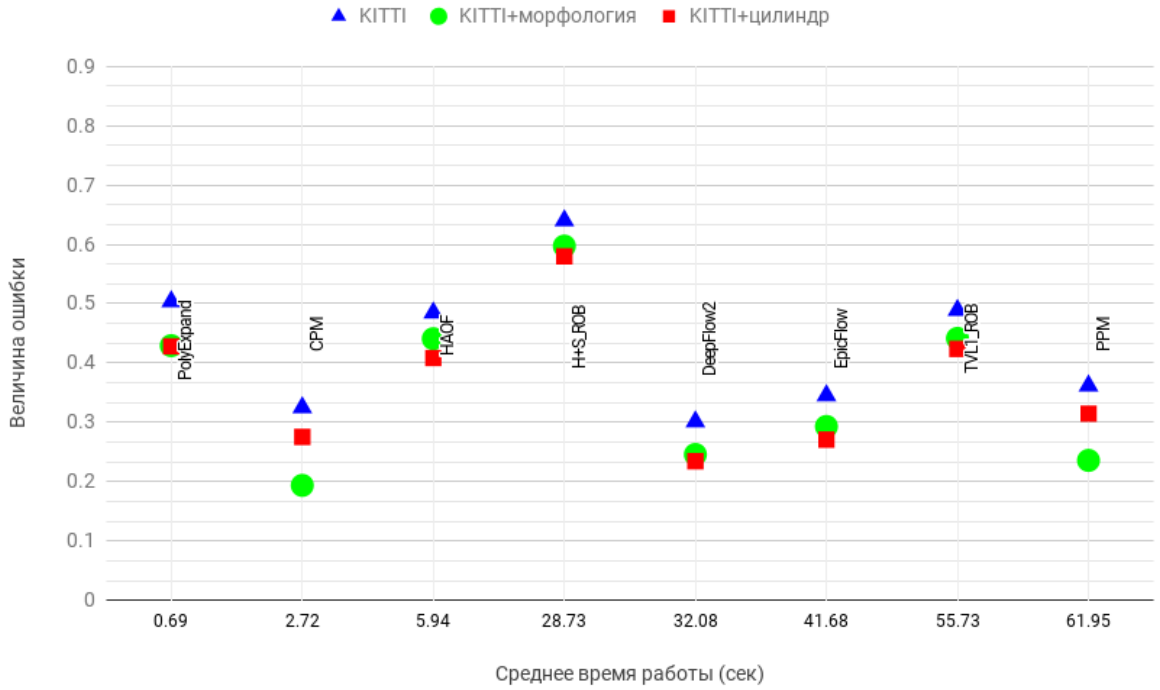


Рис. 5. График средней величины ошибки от среднего времени работы алгоритмов с интерполяцией вычисленных оптических потоков и с учетом окклюзионных пикселей. Треугольниками обозначены значения по метрике KITTI, кругами – KITTI-морфологии, квадратами – KITTI-цилиндра.



3. ОЦЕНКА КАЧЕСТВА ОПТИЧЕСКОГО ПОТОКА ПО КАРТЕ ГЛУБИНЫ: ОБСУЖДЕНИЕ

Согласно работе [3] для оценки качества алгоритмов стереозрения и вычисления оптического потока в контексте решения задач обнаружения препятствий и ухода от столкновений нужно проводить анализ соответствующей им карты глубины. Это позволяет оценивать ошибку не только с точки зрения направления наблюдения за объектом в поле зрения камеры, но и принимать во внимание расстояние до него: если глубина на близком объекте оценена ошибочно это хуже, нежели ошибка на далёком объекте.

Исследуя возможность реализации этого подхода, мы воспользовались библиотекой COLMAP [6], которая позволяет решить задачу восстановления глубины. Предварительные результаты восстановления глубины с помощью различных алгоритмов, представлен на рисунках 6. Практика показала, что COLMAP плохо справляется с вычислением глубины на сильно удалённых точках (в центре изображения), плюс к этому, неточности его работы приводят к несистематическому прореживанию данных, уменьшая достоверность полученных оценок. Ввиду вышесказанного, в отсутствии более быстрого программного решения с хорошим качеством, целесообразность развития такого подхода для оценки ОП находится под вопросом.

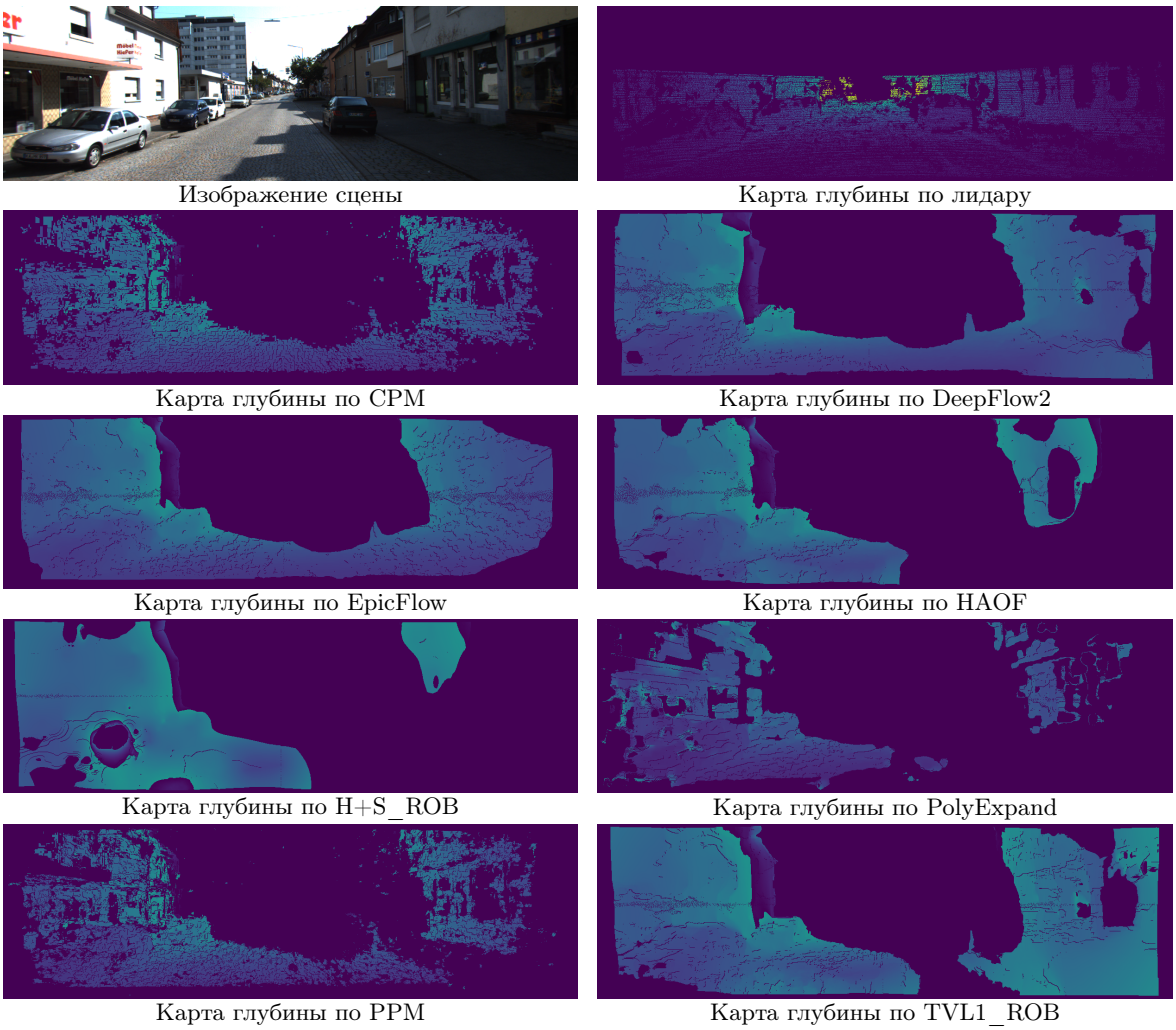


Рис. 6. Сцена, истинная карта глубины, результаты вычисления глубины по оптическому потоку. Камера движется прямолинейно.

4. ЗАКЛЮЧЕНИЕ

В этой работе мы исследовали общепринятую метрику KITTI для оценки качества оптического потока. Поскольку метрика определяет только долю плохих пикселей – она нечувствительно фактическому их распределению по области определения изображения. В результате было исследовано две модификации метрики KITTI, позволяющие устранить этот недостаток, и показано, что согласно этим метрикам упорядочение алгоритмов по точности меняется. Особенно разница в упорядочении заметна при учете плотности ошибок вычисления оптического потока, что существенно в контексте задач обнаружения препятствий.

Показано, что время работы алгоритмов на одном вычислителе существенно отличается от заявленного, что критично при выборе алгоритма как части некоторой системы технического зрения с ресурсными и временными ограничениями.

СПИСОК ЛИТЕРАТУРЫ

1. Geiger A., Lenz P., Stiller C., Urtasun R. “Vision meets robotics: The KITTI dataset”. *The International Journal of Robotics Research*, 2013, vol. 32, no. 11, pp. 1231-1237

2. Scharstein D., and Szeliski D. “Middlebury stereo vision page”, 2002.

3. Sidorchuk D., Gusamutdinova N., Konovalenko I., Ershov E. “Problem-oriented stereo vision quality evaluation complex.” *International Society for Optics and Photonics*. Eighth International Conference on Machine Vision (ICMV 2015). vol. 9875, p. 98751K. 2015.

4. Смагина А. А., Шепелев Д. А., Ершов Е. И., Григорьев А. С. “Качество детектирования препятствий как проблемно-ориентированный подход к оценке стереоалгоритмов в задачах распознавания дорожной ситуации.” *Информационные технологии и нанотехнологии*. 2018, pp. 1233-1242.

5. Tippetts B., Lee Dah Jye, Lillywhite K., Archibald J. Review of stereo vision algorithms and their suitability for resource-limited systems. *Journal of Real-Time Image Processing*. Springer. 2016. vol. 11, no. 1, pp. 5–25.

6. Schönberger, Johannes Lutz and Frahm, Jan-Michael “Structure-from-Motion Revisited” *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

7. Yinlin Hu and Rui Song and Yunsong Li, “Efficient Coarse-to-Fine PatchMatch for Large Displacement Optical Flow” *CVPR*. IEEE. 2016.

8. Fangjun Kuang, “PatchMatch algorithms for motion estimation and stereo reconstruction” *University of Stuttgart, Germany*. 2017.

9. Revaud, Jerome and Weinzaepfel, Philippe and Harchaoui, Zaid and Schmid, Cordelia, “EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow” *CVPR 2015 - IEEE Conference on Computer Vision & Pattern Recognition*. 2015.

10. Weinzaepfel, Philippe and Revaud, Jerome and Harchaoui, Zaid and Schmid, Cordelia, “DeepFlow: Large displacement optical flow with deep matching” *IEEE Intenational Conference on Computer Vision (ICCV)*. 2013.

11. Thomas Brox and Andrés Bruhn and Nils Papenberg and Joachim Weickert “High accuracy optical flow estimation based on a theory for warping” *ECCV*. 2004.

12. Sánchez Pérez, Javier and Meinhardt-Llopis, Enric and Facciolo, Gabriele “TV-L1 Optical Flow Estimation” *Image Processing On Line*. 2013. vol. 3, pp. 137–150.

13. Meinhardt-Llopis, Enric and Sánchez Pérez, Javier and Kondermann, Daniel “Horn-Schunck Optical Flow with a Multi-Scale Strategy” *Image Processing On Line*. 2013. vol. 3, pp. 151–172.

About assessing the quality of optical flow obstacle detection

A. A. Smagina, D.A. Shepelev and E.I. Ershov

Abstract—The question of choosing a dense optical flow (OF) algorithm for obstacle avoidance in real time is considered. To assess the quality of the algorithms we used the KITTI metric and also proposed its modifications to account for the distribution bad pixels. Computational experiments have shown that the ordering of algorithms for these metrics is different, which can be significant in the context of obstacle avoidance. As a result we obtained a set of operational characteristics of the OF algorithms.

KEYWORDS: Optical flow, obstacle avoidance, KITTI

.

KEYWORDS: optical flow, obstacle detection, collision avoidance, KITTI