

Санкт-Петербургский политехнический университет Петра Великого
Институт прикладной математики и механики
Высшая школа прикладной математики и вычислительной физики

Обработка и интерпретация сигналов

Решение задачи с семантическим разрывом

Работу
выполнила:
Т. В. Алпатова
Группа:
3630102/70201
Преподаватель:
В. И. Кацман

Санкт-Петербург
2021

Содержание

1. Постановка задачи	3
2. Требования к исходной фотографии	3
3. Dataset	3
4. Общий план решения	4
4.1. Определение главных точек арки	4
4.2. Определение главных точек красной коробки	4
4.3. Получение ответа по главным точкам коробки и арки	5
5. Итерации	8
5.1. Изначальный план	8
5.2. Итерация 1	9
5.2.1. Определение главных точек арки	9
5.2.2. Определение главных точек коробки	10
5.2.3. Получение ответа по главным точкам коробки и арки	13
5.2.4. Весь алгоритм	15
5.3. Итерация 2	16
6. Результаты работы	17

1. Постановка задачи

Пусть дано изображение, на котором изображено два объекта:

- объект А – арка, сделанная из картона
- объект Б – коробка в форме параллелепипеда

Задача: определить с точностью до 5 мм пролезет ли Б под арку А, если перемещать предметы можно только параллельным переносом и не отрывая от поверхности, на которой они стоят

2. Требования к исходной фотографии

- Фотографии квадратные
- Фотографии подаются на вход в формате *.jpg
- Фотографии сделаны на камеру телефона с разрешением 12 мегапикселей
- Фотографии не смазаны и не размыты.
- Оба объекта должны присутствовать на фотографии
- Кроме объектов А и Б на фотографии не должно быть других предметов
- Объекты четко видны на фотографии
- Предметы не могут перекрывать друг друга полностью или частично
- Границы предметов не должны выходить за рамки фотографии
- Не допускается съемка с точки зрения, при которой не видно отверстие в объекте А (арка).
- Объекты освещены с помощью настольного светильника и/или комнатного света. Светильник стоит в определенном положении на столе. Это положение должно позволять делать четкие, не засвеченные и не темные фотографии.
- Точка съема должна выбираться так, чтобы были видны три плоскости каждого объекта. Например, не допускаются фотографии, сделанные сверху, или фотографии с уровня стола.
- На фотографии присутствуют две плоскости - стена и стол. Допускается присутствие третьей плоскости - плоскости рядом стоящего шкафа, при условии, что ее почти невидно.

3. Dataset

Dataset содержит две папки "Да" и "Нет". Название папки - ответ на вопрос задачи. Dataset доступен по ссылке <https://drive.google.com/drive>

4. Общий план решения

В изначальном плане и в последующих итерациях основные шаги не менялись. Менялась их реализация.

4.1. Определение главных точек арки

После выполнения этого пункта мы должны знать координаты точек, отмеченных на рисунке:

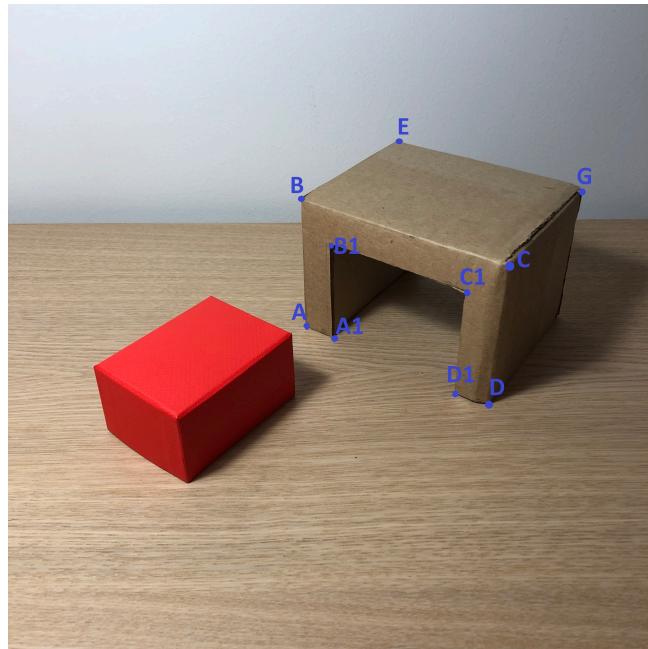


Рисунок 1. Главные точки арки

4.2. Определение главных точек красной коробки

Для коробки нам нужно знать по две главные точки для высоты и ширины. На каждой картинке показаны пары точек, координаты которых нам нужно узнать (достаточно только одной пары точек)

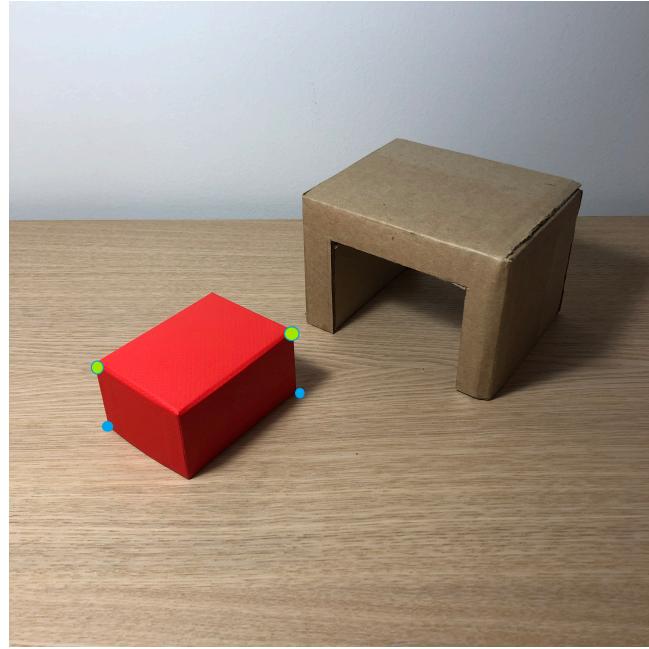


Рисунок 2. Главные точки коробки по ширине

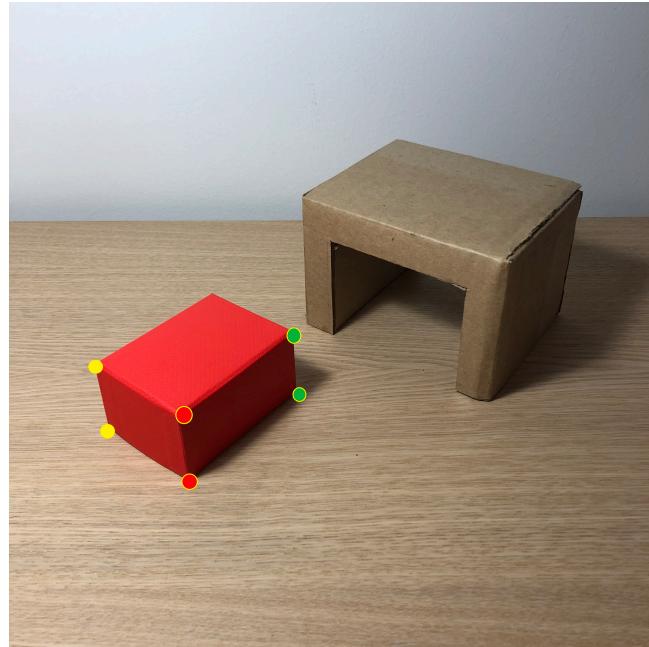


Рисунок 3. Главные точки коробки по высоте

4.3. Получение ответа по главным точкам коробки и арки

Данный пункт основан на идеи того, как человек видит изображения. Параллельные линии в жизни сходятся в одну точку.

Идейно этот пункт не менялся в ходе решения задачи.

Условие пролезания коробки под арку: коробка пролезет под арку, если проекция коробки на плоскость, которая содержит отверстие арки, не перекрывает область отверстия арки.

Рассмотрим подробнее, как проверить это условие.

- **Фокусы.** Находим две точки фокуса - те точки, в которых сходятся параллельные линии.

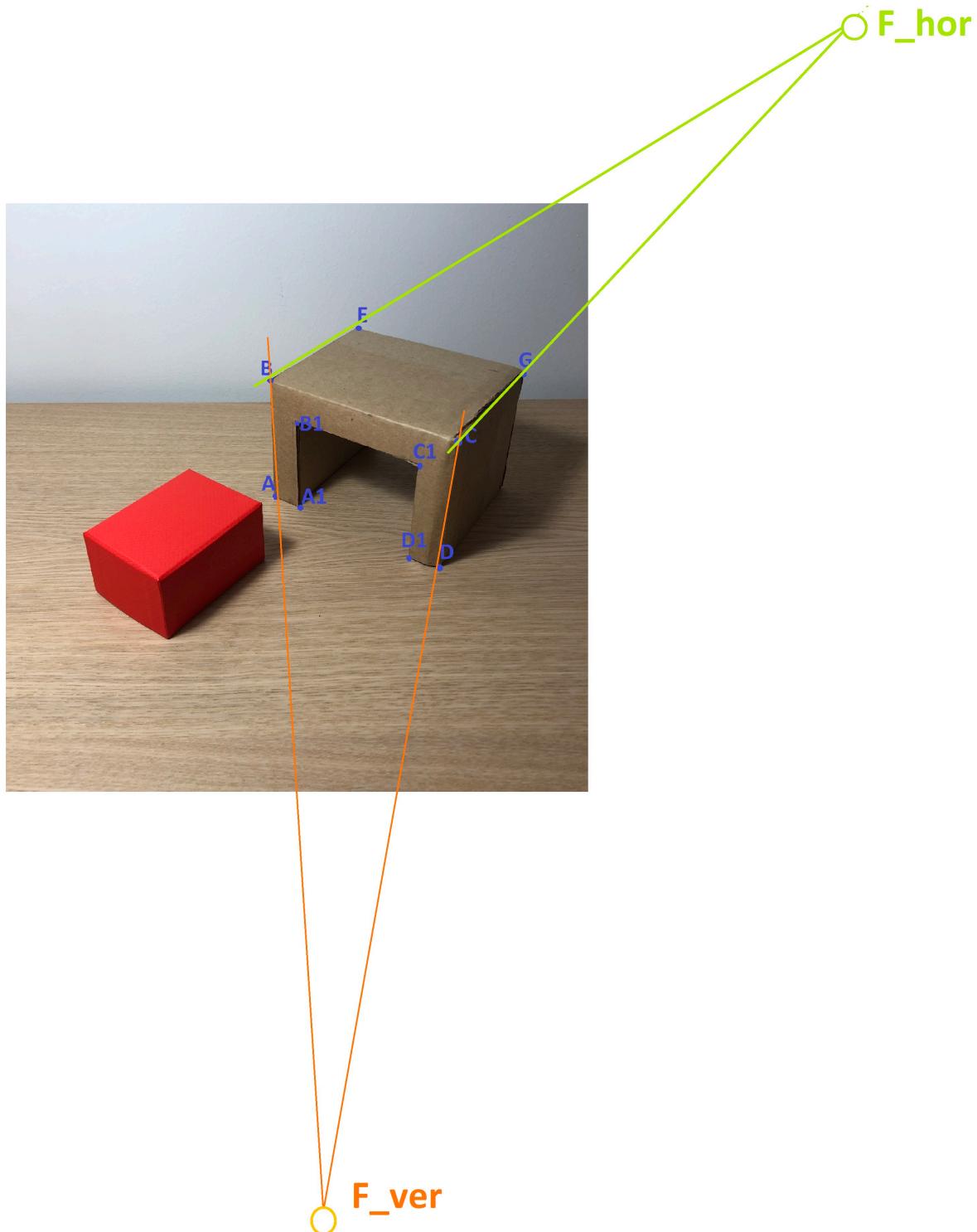


Рисунок 4. Точки фокуса арки

- **Определение, пройдет ли коробка по высоте.** Находим точку p_2 , с помощью построений, показанных на рисунке

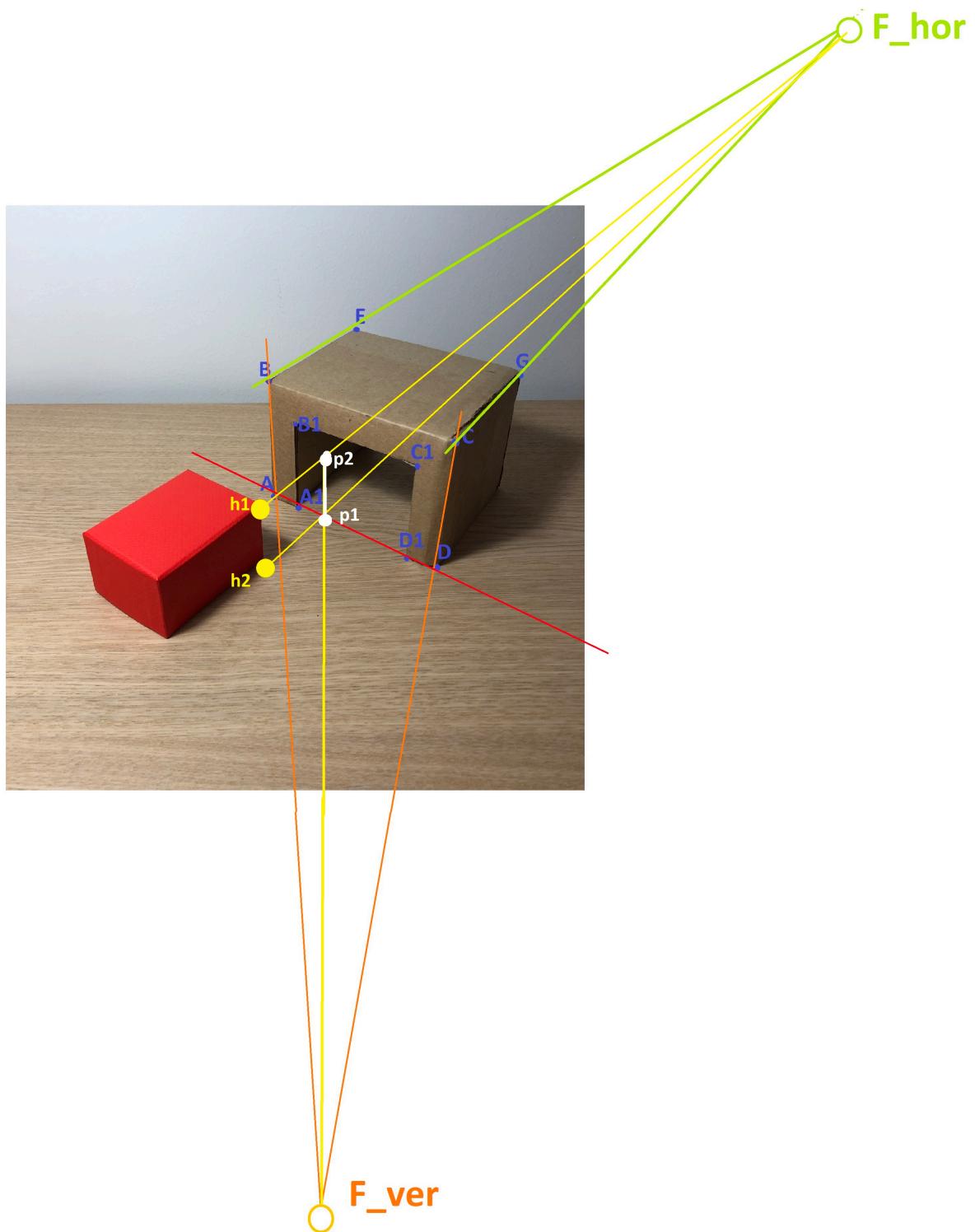


Рисунок 5. Проекция высоты коробки на плоскость арки

Если p_2 лежит выше, чем прямая b_1c_1 , то коробка не пройдет по высоте. Иначе - пройдет

- **Определение, пройдет ли коробка по ширине.** Находим точки p_1, p_2 , с помощью построений, показанных на рисунке:

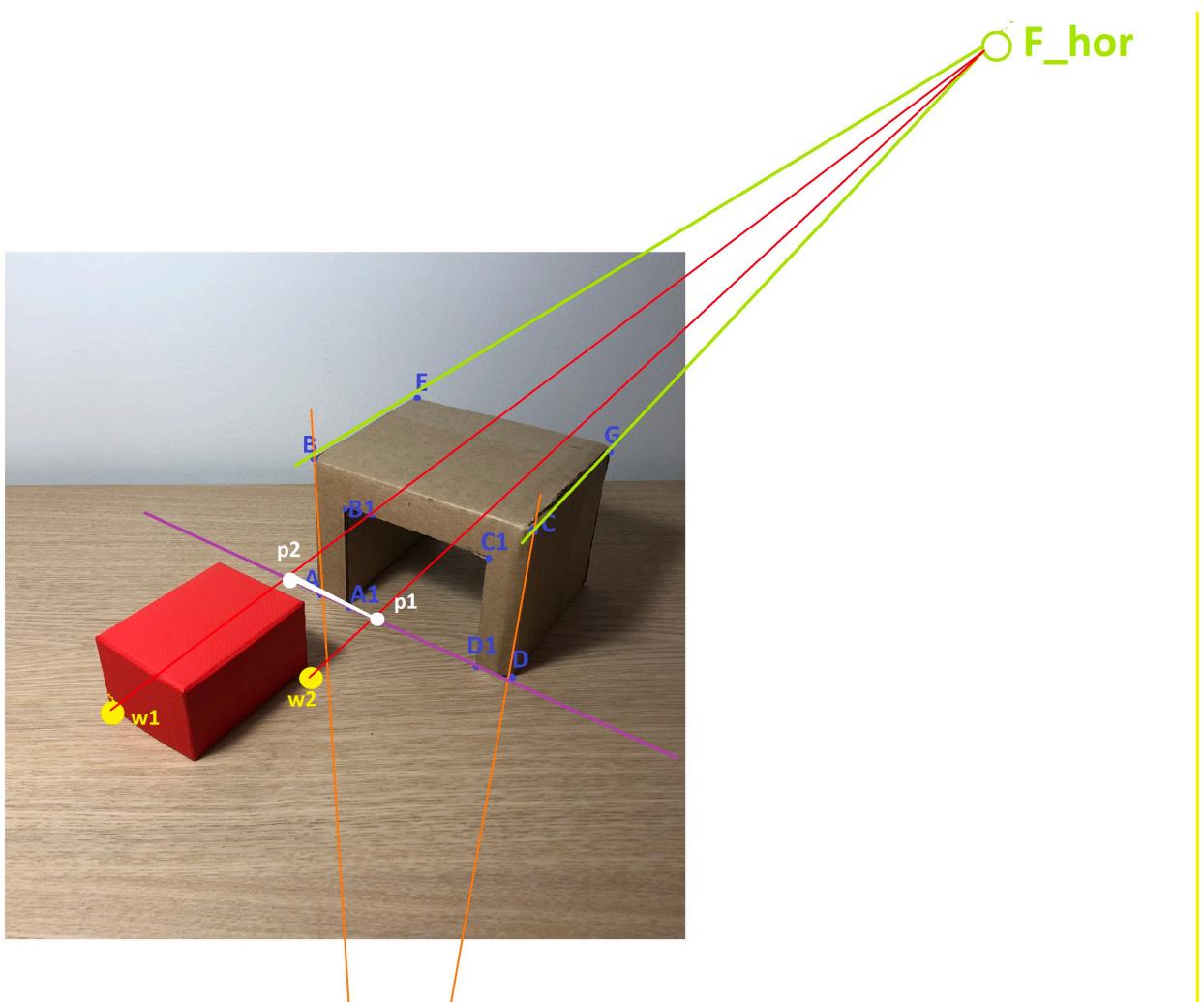


Рисунок 6. Проекция ширины коробки на плоскость арки

Если расстояние между p_1 и p_2 больше расстояния между a_1 и d_1 , то коробка не пройдет по ширине.

- **Ответ:** да, если коробка проходит и по высоте и по ширине. Иначе - нет.

5. Итерации

5.1. Изначальный план

Изначальный план решения был таков:

1. Ищем границы детектором Canny
2. С помощью алгоритма Хаффа находим линии на границах, определенных на предыдущем шаге.
3. Линии, которые обрамляют границы коробки, легко отделить, так как коробка имеет ярко выраженный красный цвет.
4. Ищем точки пересечения линий коробки.

5. Ищем точки пересечения оставшихся линий. Зная форму арки, делаем вывод о местоположении определенных точек на арке.

Данный план потерпел крушение сразу же при попытке его реализовать. возникли следующие проблемы:

- При поиске границ детектором Canny, находятся линии текстуры стола
- При увеличении σ в Гауссиане, текстуры стола сглаживаются, но вместе с тем сглаживаются и объекты. Линии на них не определяются.
- Придумать алгоритм для программного определения местоположения точек арки оказалось не так просто.

5.2. Итерация 1

На первой итерации было предпринято много попыток устраниить проблемы изначальной идеи. Опишем их

- Отсеять текстуру стола. Для этого было подготовлено несколько шаблонов текстуры стола. Для этого шаблона и каждого кусочка изображения вычислялась GLCM. Далее эти матрицы сравнивались и делался вывод о сходстве текстуры. По результаты сравнения получалась маска. Минус подхода в том, что алгоритм часто путал текстуру стола и с текстурой арки, а так же иногда не мог определить текстуру стола. Были опробованы различные параметры, по которым сравнивалось (contrast, dissimilarity, homogeneity, correlation), а так же различные пороги для сравнения
- Для решения той же проблемы так же был опробован алгоритм Local Binary Pattern. Его применение так же не дало положительных результатов.
- После неудачных попыток отсеять стол, была принята попытка использовать детектор особых точек. По особым точкам определялась бы матрица гомографии, на основе которой получались бы точки для определения ответа. Проблема опять же была в том, что алгоритм цеплялся за особые точки стола (так как он имеет характерный рисунок) и неправильно определял преобразования. Данный алгоритм не удалось применить к красной коробке, однако для арки применение нашлось

Теперь перейдем к тому, как же был реализован алгоритм

5.2.1. Определение главных точек арки

Как уже было сказано, для определения главных точек стола применялся механизм поиска особых точек. В результате экспериментов, было получено, что наилучший результат получается при поиска особых точек детектором ORB и сравнении дескрипторов точек на основе метрики Хэмминга.

Так же было обнаружено, что одного шаблона недостаточно для определения особых точек на всех изображениях. Арка на одних изображениях повернута правым боком, на других левым. Некоторые изображения сделаны с более высокой позиции, другие более с низкой.

Было принято решение подговорить словарь шаблонов. Для подготовки словаря было сделано около десятка фотографий арки в разных ракурсах. На этих изображениях вручную были найдены координаты точек. Далее с помощью детектора ORB были найдены особые точки. Вся эта информация записывалась в файл.

Когда изображение поступает на вход алгоритму, он вычисляет его особые точки. Далее ищет в словаре наиболее похожий шаблон. За оценку схожести шаблонов принимается среднее значение расстояния между сопоставленными дескрипторами точек. Чем эта метрика меньше, тем более похожи особые точки. Далее на основе этого наилучшего шаблона находится матрица гомографии, а по ней уже перспективные преобразования, на основе которых получаются координаты нужных точек.

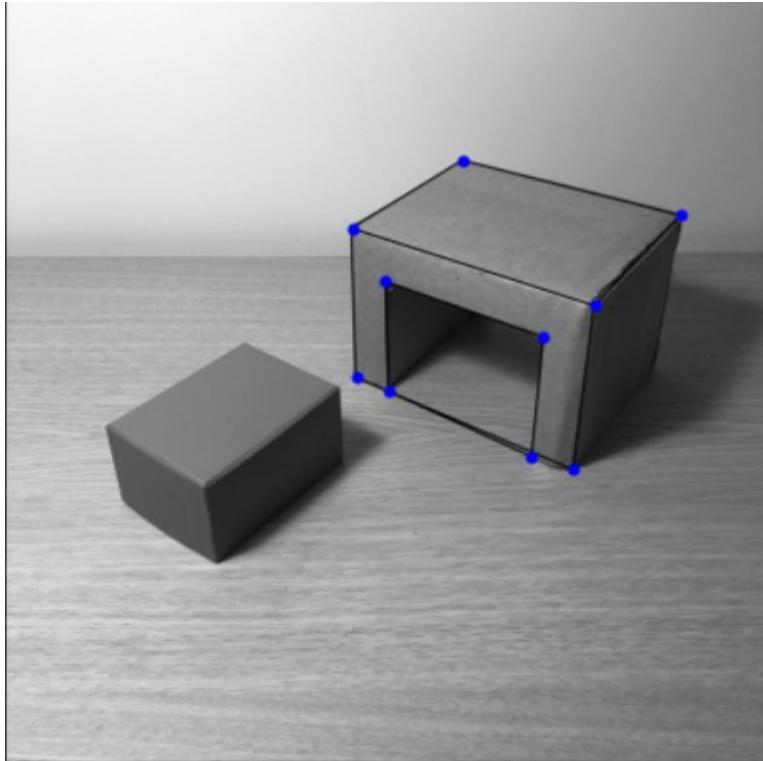


Рисунок 1. Пример работы алгоритма на шаге 1

5.2.2. Определение главных точек коробки

По скольку цвет коробки выделяется от всех остальных цветов изображения, то было решено найти какую-то маску, которая могла бы отделить область, в которой находится коробка. В результате экспериментов была получено следующая маска:

$$mask = (image_blue - image_gray <= 70)$$

При использовании такой маски часто действовалась область, где находится арка. Но поскольку ее главные точки к этому шагу уже известны, то область маски, в которой была обнаружена арка, обнулялась.

После чего к маске применяется бинарная операция закрытие, границы маски расширяются на несколько пикселей и маска делается выпуклой (с помощью функции *convex_hull_image*)

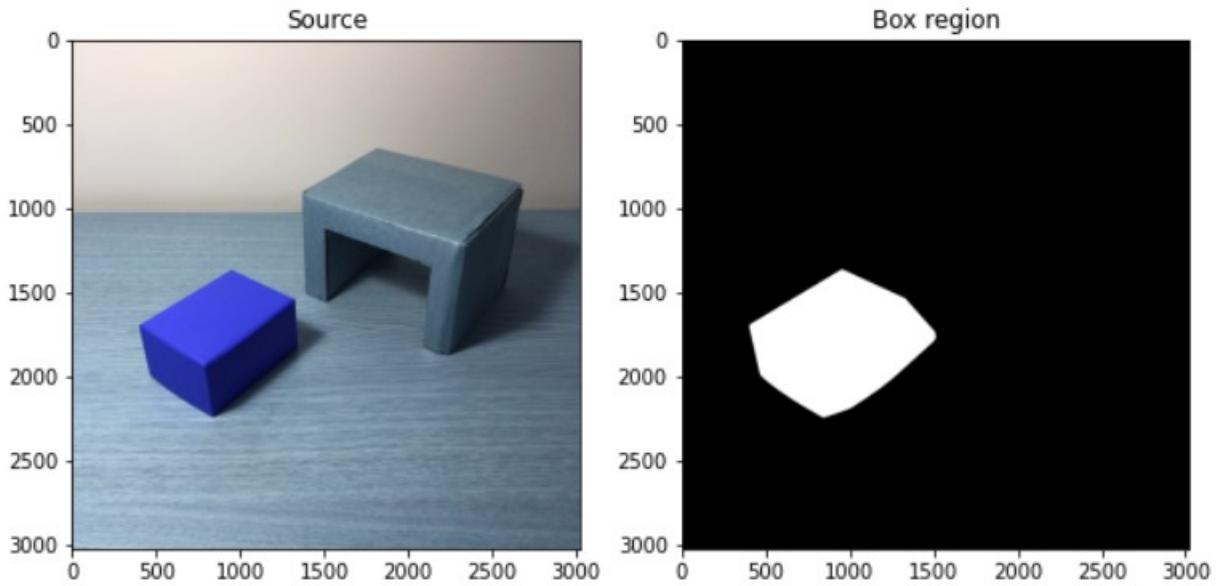


Рисунок 2. Пример нахождения маски коробки

Далее методом Canny ищутся границы на изображении. Было попробовано предварительное сглаживание изображения фильтром Гаусса или Собеля. В результате чего фильтр Гаусса показал себя лучше.

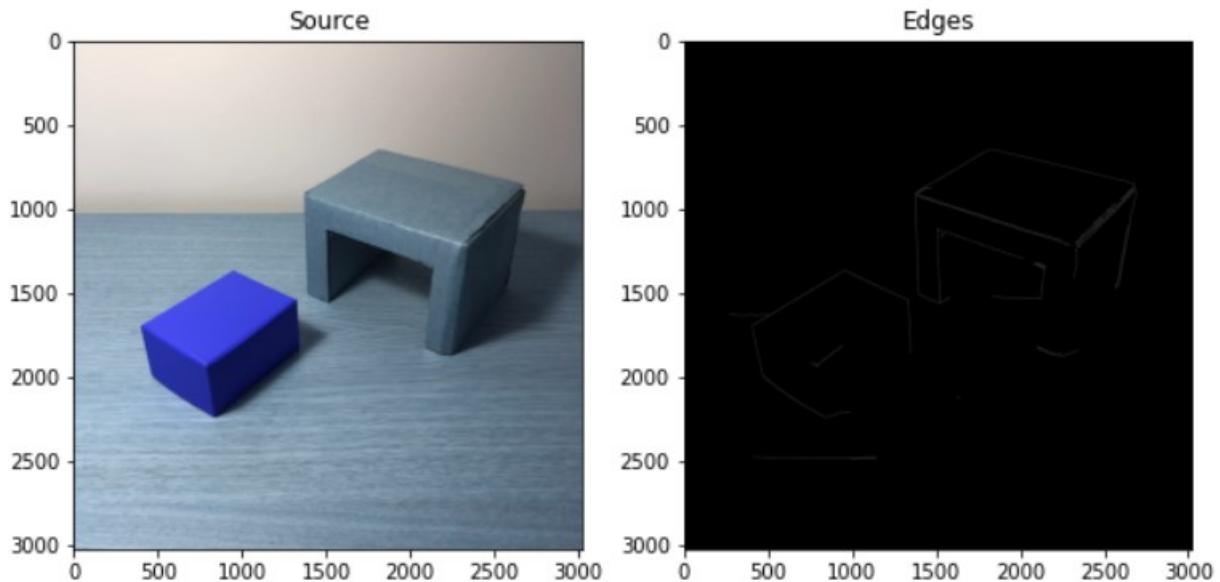


Рисунок 3. Пример нахождения границ на изображении

Далее на картинку с границами накладывается маска коробки. Она расширялась и становилась выпуклой, дабы окружающие ее линии не пропали на результирующем изображении. Таким образом получается картинка, которая содержит только границы коробки.

На изображении с границами коробки с помощью детектора Хаффа обнаруживаются прямые. Тут возникли следующие трудности:

- Нужно определить "параллельные" прямые

- Иногда в одном и том же месте обнаруживается несколько прямых
- Не все прямые обнаруживаются

Примечание: здесь и далее границы на изображениях изображены более толстыми линиями, чтобы их было четче видно

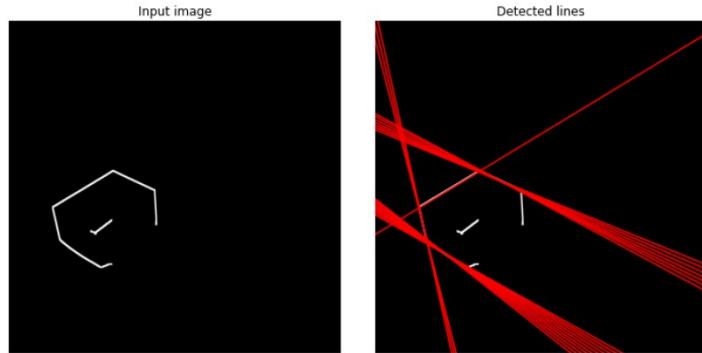


Рисунок 4. Проблемы, возникающие при нахождении прямых детектором Хаффа

Для решения проблем применялось следующее:

- Линии условно можно разделить на три группы: прямые, левые и правые. У всех у них разный угол наклона. У прямых $\text{abs}(\text{angle}) \leq 0.3$. У левых $\text{angle} > 0.3$, у правых $\text{angle} < -0.3$. Порог 0,3 был выбран исходя из экспериментов.

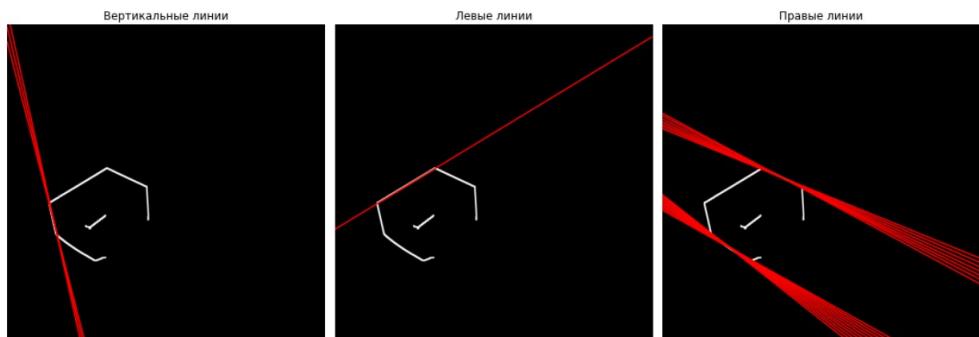


Рисунок 5. Решение проблемы 1. Сортировка линий

- У линий, которые очень похожи близкое d , поэтому из каждой группы линий мы выбираем линии так, чтобы d у них отличались

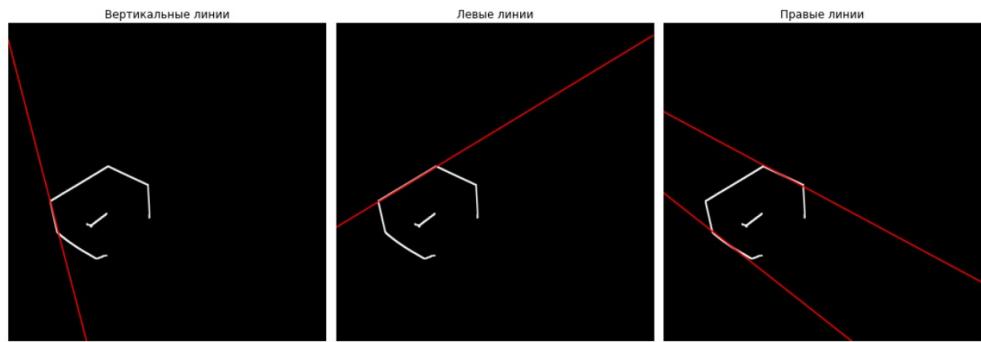


Рисунок 6. Решение проблемы 2. Обнаружение нескольких прямых в одном месте

- Нам не обязательно знать все прямые для того, чтобы получить ключевые точки. Можно найти любую пару точек и одни и те же точки можно считать как пересечение разных линий. В результате получилось следующее.

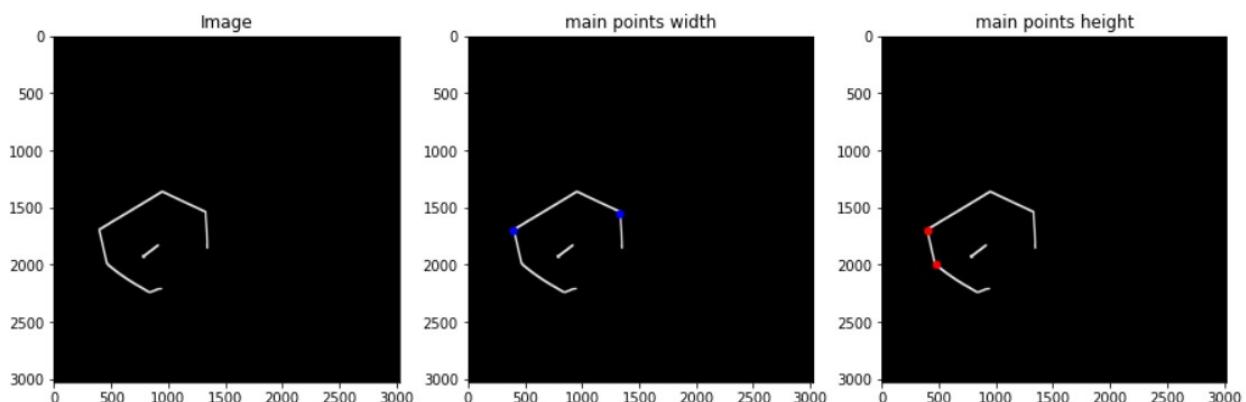


Рисунок 7. Пример нахождения особых точек коробки

5.2.3. Получение ответа по главным точкам коробки и арки

После второго шага, нам известны все точки. По ним исходя из элементарных геометрических расчетов определяется, может ли пройти коробок под арку. Приведем пример, иллюстрирующий работу алгоритма на данном шаге.

Нахождение F_ver

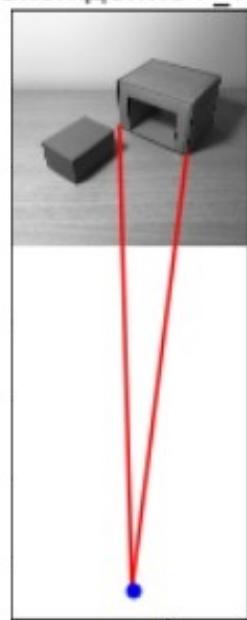


Рисунок 8. Пример нахождения вертикального фокуса

Нахождение F_hor

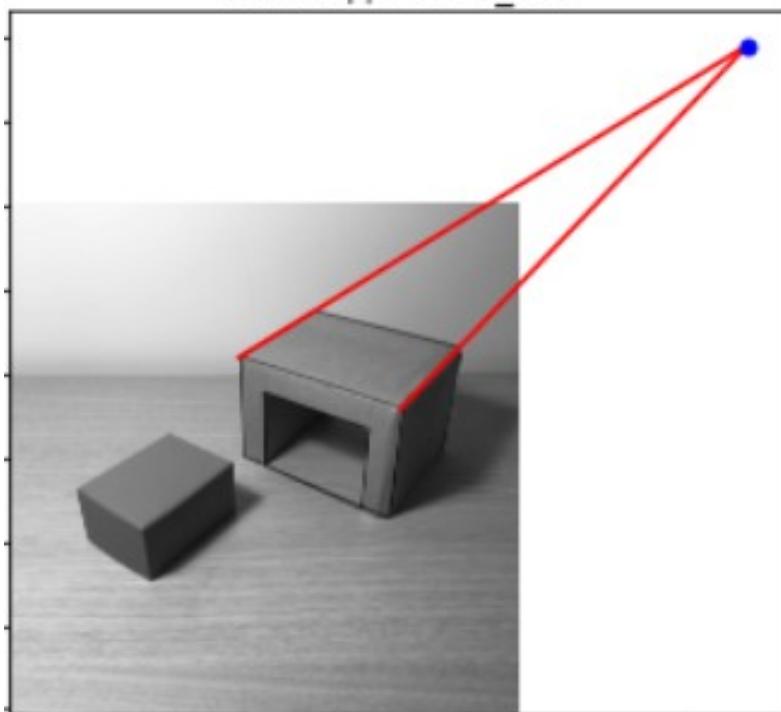


Рисунок 9. Пример нахождения горизонтального фокуса

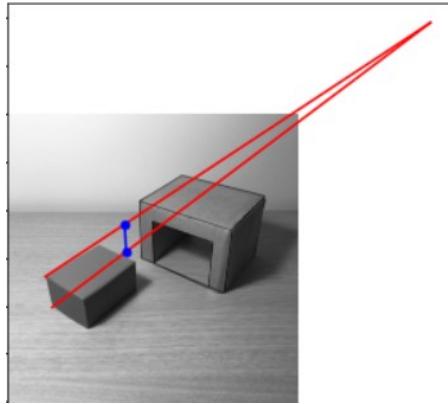


Рисунок 10. Пример определения высоты коробки на плоскость арки

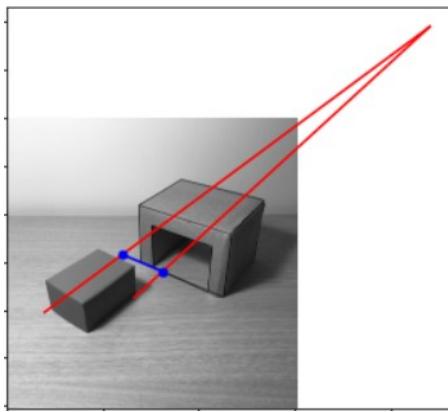


Рисунок 11. Пример определения ширины коробки на плоскость арки

5.2.4. Весь алгоритм

Резюмируя первую итерацию приведем весь алгоритм

- Определение главных точек арки
 1. Заранее составляется словарь шаблонов
 2. По входному изображению считаются дескрипторы особых точек
 3. Среди шаблонов находится наиболее похожий по дескрипторам особых точек
 4. По этому шаблону и исходному изображению считается матрица гомографии
 5. Находятся преобразования и главные точки арки
- Определение главных точек коробки
 1. Выделяются границы изображения детектором Canny
 2. Создается маска, содержащая только коробку
 3. Маска применяется к изображению с границами. Получаются только границы коробки
 4. На этом изображении ищутся линии детектором Хаффа.
 5. По линиям находятся главные точки коробки
- Получение ответа по главным точкам коробки и арки

5.3. Итерация 2

Общая идея алгоритма не изменилась по сравнению с 1 итерацией. Были лишь внесены некоторые детали, которые улучшили алгоритм.

- *Анализ цвета коробки.* Был проведен анализ цвета коробки. На первой итерации перебирались различные комбинации, благодаря которым была получена маска. Во второй итерации были рассмотрены интенсивности нескольких пикселей коробки:

Красный: 189	Красный: 139	Красный: 203
Зеленый: 54	Зеленый: 69	Зеленый: 63
Синий: 48	Синий: 59	Синий: 62
Красный: 168	Красный: 237	Красный: 64
Зеленый: 49	Зеленый: 91	Зеленый: 19
Синий: 45	Синий: 94	Синий: 14

Рисунок 12. Интенсивности пикселей коробки

В результате чего была придумана другая маска:

$$mask = (r > g) * (g - b < 20) * (g - b > -5) * (r > 70) * (g < 90)$$

По визуальным наблюдениям можно утверждать, что применение такой маски дает лучшие результаты.

- *Сортировка прямых.* Порог 0.3 при сортировке прямых был выбран исходя из экспериментов. Тем не менее, для каждого изображения этот порог уникален и применение одного порога ко всем изображением достаточно грубый подход. По сути нам нужно разбить значение углов на 3 класса. Для этого во второй итерации применяется метод *KMeans*. Далее по среднему значению угла в классе делается вывод о том, какой класс каким прямым соответствует.
- Для экономии времени сначала ищется маска коробки, а затем детектор Canny применяется не ко всему изображению, а только к области, выделяемой маской

Так же после второй итерации был проведен рефакторинг кода по результатам которого:

- Код стал структурированный. Каждая логическая часть вынесена в отдельный модуль
- Учтены замечания ревьюеров
- Написан код, который позволяет оценивать результаты на всем датасете

6. Результаты работы

Алгоритм может выдавать три ответа - да, нет, не знаю.

Ответ не знаю выдается в тех случаях, когда алгоритм не смог определить главные точки коробки. Это может произойти потому, что в алгоритме Хаффа не было найдено достаточное количество прямых для того, чтобы сделать вывод.

		Real	
		True	False
Predicted	True	20	13
	False	12	15
	None	2	0

Точность алгоритма 0.56