САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ - ПРОЦЕССОВ УПРАВЛЕНИЯ

**Инишева Дарья Олеговна**

**Магистерская диссертация**

Определение размеров торцов бревен по стерео-реконструкции и нейронной сети

Направление 1.04.02. «Прикладная математика и информатика»

Основная образовательная программа ВМ.5759 «Цифровая экономика»

Научный руководитель,

кандидат технических наук,

доцент кафедры КММС

Гришкин В.М.

Рецензент: старший научный

сотрудник ООО <НСН>,

кандидат технических наук,

Епифанов Н.А.

Санкт-Петербург 2021

**Содержание**

Введение………………………………...................................3

Постановка задачи …………………………………………..4

Актуальность задачи ………………………………………..5

Обзор литературы……………………………………………6

Глава 1. Алгоритм нахождения размеров торцов бревен по стерео-реконструкции и нейронной сети……………………7

1.1. Построение плотной стерео-реконструкции торцов бревен……………………………………………………7

1.1.1. Захват изображений с камер………………...7

1.1.2. Калибровка камеры…………………………...8

* + 1. Построение облака точек реконструкции…15
  1. Применение Нейросети для определения эллипсов торцов……………………………………………...19
  2. Алгоритм нахождения размеров торцов по нейронной сети и стерео-реконструкции……………………...25
  3. Улучшения………………………………………….29

Глава 2. Алгоритм нахождения объема штабеля по движущемуся лесовозу……………………………………….31

2.1. Захват изображений с камер……………………….31

2.2. Построение плотной реконструкции по серии

фото……………………………………………………...32

2.3. Улучшения………………………………………….43

Результаты……………………………………………………43

Заключение…………………………………………………...44

Литература……………………………………………………45

**Введение**

В наши дни компьютерное или машинное зрение это одна из самых быстро развивающихся областей науки.

**Машинное зрение** — это научное направление в области искусственного интеллекта, в частности робототехники, и связанные с ним технологии получения изображений объектов реального мира, их обработки и использования полученных данных для решения разного рода прикладных задач без участия (полного или частичного) человека. В наши дни множество процессов на производстве и в обычной жизни, которые на данный момент выполняются людьми, можно заменить алгоритмами машинного зрения, чтобы облегчить работу людей. Машинное зрение можно использовать в следующих областях:

1. Системы управления процессами ([промышленные роботы](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BC%D1%8B%D1%88%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D1%80%D0%BE%D0%B1%D0%BE%D1%82), [автономные транспортные средства](https://ru.wikipedia.org/w/index.php?title=%D0%90%D0%B2%D1%82%D0%BE%D0%BD%D0%BE%D0%BC%D0%BD%D0%BE%D0%B5_%D1%82%D1%80%D0%B0%D0%BD%D1%81%D0%BF%D0%BE%D1%80%D1%82%D0%BD%D0%BE%D0%B5_%D1%81%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B2%D0%BE&action=edit&redlink=1)).
2. [Системы видеонаблюдения](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%B2%D0%B8%D0%B4%D0%B5%D0%BE%D0%BD%D0%B0%D0%B1%D0%BB%D1%8E%D0%B4%D0%B5%D0%BD%D0%B8%D1%8F).
3. Системы организации информации (например, для индексации баз данных изображений).
4. Системы моделирования объектов или окружающей среды (анализ медицинских изображений, топографическое моделирование).
5. Системы взаимодействия (например, устройства ввода для системы человеко-машинного взаимодействия).
6. Системы [дополненной реальности](https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%BF%D0%BE%D0%BB%D0%BD%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D1%80%D0%B5%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C).
7. [Вычислительная фотография](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%84%D0%BE%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F), например, для мобильных устройств с камерами.

Одним из направлений машинного зрения является стереозрение. Стереозрение — это извлечение трехмерной информации из цифровых изображений, например, полученных с помощью камеры. Оно позволяет получить представление о глубине изображения и расстоянии до объектов, составить трехмерную картину окружающего мира.

**Постановка задачи**

Необходимо придумать и реализовать алгоритм по определению размеров торцов бревен, который бы сочетал в себе алгоритмы стереозрения и нейронные сети. А также придумать комбинацию известных решений Structure from motion для определения объема проезжающего штабеля. Оба алгоритма будут работать как без участия человека, так и без использования эталона – объекта, размеры которого заранее известны.

Задачу определения размеров торцов бревен по стерео-реконструкции и нейронной сети можно разбить на 3 подзадачи:

1. Применение алгоритмов стереозрения к кадрам штабеля лесовоза с разных камер для получения трехмерной реконструкции штабеля.

2. Применение нейронной сети к кадрам штабеля лесовоза для получения боксов, в которых содержатся торцы бревен.

3. Применение алгоритма по нахождению реальных размеров торцов на основе работы нейронной сети и алгоритмов стереозрения.

Выходами всего алгоритма будут являться размеры большой и малой осей эллипсов для всех торцов.

Второй алгоритм по подсчету объема штабеля лесовоза, проезжающего мимо металлической рамки, использует иной подход:

1. Использование алгоритма Structure from motion.
2. Восстановление реконструкции всего штабеля лесовоза из изображений/видео движущегося лесовоза.
3. Определение размеров штабеля по плотной реконструкции методами из Opencv.

**Актуальность задачи**

На сегодняшний день у задач стереозрения множество различных применений, особенно часто стереозрение применяется в области робототехники, так как передвижение роботов и определение расстояния до объектов осуществляется с помощью алгоритмов стереозрения.

Если рассматривать данную конкретную задачу - определение размеров торцов бревен и определение объема штабеля, то представленное решение будет актуально на множестве производств, работающих с лесом, в частности, у которых производится приемка леса. На данный момент на таких предприятиях измерение объема леса происходит с помощью человека, а именно: человек делает замеры нескольких бревен в лесовозе, а затем визуально подсчитывает их количество. После этого, берется среднее по измеренным бревнам и умножается на примерное количество бревен. Неточность оценки в данном случае может достигать 30%. Очевидно, что алгоритмы стереозрения справятся с этой задачей и точнее и быстрее.

Также, уже существуют алгоритмы измерения диаметров торцов бревен по фотографии, но данные методы используют измерение эталонных объектов на фотографии. Эталонным объектом может выступать любой объект, размер которого известен. Измеряя длину эталона на фотографии и сравнивая ее с длиной эталона в действительности, определяется масштаб изображения. После этого, появляется возможность измерить остальные объекты на фотографии и привести их к реальному масштабу. Данный подход довольно прост в реализации и не требует высоких вычислительных затрат, однако, он не исключает участие человека, а также возникают трудности в установке эталона, например на проезжающем лесовозе.

**Обзор литературы**

Следует отметить популярность темы машинного зрения и нейронных сетей, благодаря этому существует большое количество источников в академической литературе.

Первые источники описывают основы машинного зрения и, в частности, стереозрения: в [1] и [2] приведена вся базовая теория по компьютерному зрению. В [3] аналогично предыдущим книгам приведена базовая информация о компьютерном зрении, но в данной книге делается акцент на применении теоретических знаний с помощью библиотеки OpenCV.

В [4], [5], [7], [11] рассматриваются особенности работы с камерами типа Kinect, а именно о принципах ее работы, специфике калибровки.

В [6], [8] подробно описываются принципы проективной геометрии, которые очень помогают в понимании того, как камера делает снимки, как ее калибровать и как получать точки системы координат камеры в мировой системе.

В [9], [10], [12] описывается история сверточных нейронных сетей с полным описанием архитектур, а именно: какие проблемы возникали в процессе развития сетей, каким образом данные проблемы решались, и какие архитектуры имеются в настоящее время. Также описана архитектура сети RetinaNet с базовой сетью ResNet, к1оторая используется в данной работе.

В [13] описывается популярный алгоритм SIFT выделения дескрипторов на изображении. В данной работе SIFT-алгоритм применяется как первый шаг пайплайна для построения разряженной реконструкции – COLMAP, в [18] же описан сам пайплайн.

В [14] описан известный алгоритм RANSAC, реализация которого применяется в множестве решений: в данном случае RANSAC используется в пайплайне COLMAP, а также в алгоритме поиска плоскости торца. В свою очередь в алгоритме RANSAC используются такие методы, как SVD разложение и Ортогональная регрессия, которые описаны в [15] и [16] соответственно.

Также для улучшения результатов алгоритма по нахождения размеров торцов была использована фильтрация, которая описана в [19], а также размытие изображение, которое можно найти в [17].

Для алгоритма нахождения объема штабеля лесовоза, была использована функция findBoundingBox, принципы работы которой описаны в [20], а также для улучшения результатов был использован алгоритм вычитания фона, который можно найти в [21].

**Глава 1. Алгоритм нахождения размеров торцов бревен по стерео-реконструкции и нейронной сети**

* 1. **Построение плотной стерео-реконструкции торцов бревен**

Алгоритм построения плотной стерео-реконструкции торцов бревен состоит из нескольких этапов. Рассмотрим каждый из них:

* + 1. **Захват изображений с камер**

Пусть имеются три pinhole камеры. Pinhole камера – фотографический аппарат без объектива, роль которого выполняет малое отверстие в камере. Данные камеры располагаются на металлической рамке высотой и длиной в один метр. Все три камеры расположены на верхней планке описанной рамки. Далее, камеры подключаются к портативному устройству, например, ноутбуку, и происходит одновременный захват изображений со всех трех камер с помощью встроенного метода *Opencv::VideoCapture().* Металлическая рамка с pinhole камерами представлена на Рис.1.



**Рис.1.** Металлическая рамка с pinhole камерами.

* + 1. **Калибровка камеры**

Прежде чем переходить к калибровке камеры, предлагается рассмотреть некоторые понятия из проективной геометрии. [8]

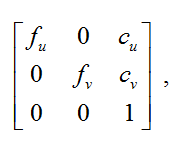
Рассматривается двухмерное проективное пространство (которое также называется проективной плоскостью). В то время как на обычной Евклидовой плоскости точки описываются парой координат, на проективной плоскости точки описываются трехкомпонентным вектором . При этом для любого ненулевого числа α, векторы и соответствуют одной и той же точке. А нулевой вектор не соответствует никакой точке и выкидывается из рассмотрения. Такое описание точек плоскости называется однородными координатами (homogeneous coordinates).

Точкам проективной плоскости можно сопоставить точки обычной Евклидовой плоскости. Координатному вектору при  сопоставляется точка Евклидовой плоскости с координатами Если же , то есть координатный вектор имеет вид , то эта точка в бесконечности. Таким образом, проективную плоскость можно рассматривать как Евклидовую плоскость, дополненную точками из бесконечности.

Перейти от однородных координат к обычным Евклидовым можно путем деления координатного вектора на последнюю компоненту и последующего ее отбрасывания → . А от Евклидовых координат  перейти к однородным можно за счет дополнения координатного вектора единичкой: →

**Прямые на проективной плоскости.** Любая прямая на проективной плоскости описывается, подобно точке, трехкомпонентным вектором  Вектор, описывающий прямую, определен с точностью до ненулевого множителя. При этом уравнение прямой будет иметь вид: .  
В случае, когда имеется аналог обычной прямой  А вектор соответствует прямой лежащей в бесконечности.  
  
**Трехмерное проективное пространство.** По аналогии с проективной плоскостью, точки трехмерного проективного пространства определяются четырехкомпонентным вектором однородных координат →. Для любого ненулевого числа α, координатные вектора  и соответствуют одной и той же точке.  
  
Как в случае проективной плоскости, между точками трехмерного Евклидова пространства и трехмерного проективного пространства можно установить соответствие. Вектору однородных координат при  соответствует точка Евклидова пространства с координатами . А точка с вектором однородных координат вида лежит в бесконечности.  
  
**Проективное преобразование.** С геометрической точки зрения, проективное преобразование — это обратимое преобразование проективной плоскости (или пространства), которое переводит прямые в прямые. В координатах, проективное преобразование выражается в виде невырожденной квадратной матрицы *H*, при этом координатный вектор *x* переходит в координатный вектор  по следующей формуле:

**Калибровка камеры** — это задача получения внутренних и внешних параметров камеры по имеющимся фотографиям или видео, отснятыми ею, а также коэффициенты дисторсии.[1] На выходе калибровки камеры необходимо получить матрицу внутренних параметров камеры, состоящую из фокусного расстояния - расстояние между оптическим центром объектива и матрицей фотоаппарата, углов наклона пикселей, принципиальной точки – точка пересечения оптической оси с плоскостью изображения, в идеальной камере находится точно в центре изображения, в реальных немного смещена от центра. Матрица камеры представлена на Рис.2.



**Рис.2.** Матрица внутренних параметров камеры, где – фокусное расстояние, измеренное в ширине и высоте пикселей, – координаты принципиальной точки.

Помимо внутренних параметров камеры, необходимо получить коэффициенты дисторсии. Существуют два основных вида дисторсии: радиальная дисторсия и тангенциальная дисторсия.  
  
Радиальная дисторсия — искажение изображения в результате неидеальности параболической формы линзы. Искажения, вызванные радиальной дисторсией, равны 0 в оптическом центре сенсора и возрастают к краям. Как правило, радиальная дисторсия вносит наибольший вклад в искажение изображения.  
  
Тангенциальная дисторсия — искажения изображения, вызванные погрешностями в установки линзы параллельно плоскости изображения.

Для устранения дисторсии координаты пикселей можно пересчитать с помощью следующих уравнений:

где — первоначальное расположение пикселя,  
(*,*) — расположение пикселя после устранения геометрических искажений,— коэффициенты радиальной дисторсии,  
— коэффициенты тангенциальной дисторсии,

Точность измерения параметров камеры (коэффициенты дисторсии, матрица камеры) определяется средней величиной ошибки репроецирования— расстояние (в пикселях) между проекцией *P'* на плоскость изображения точки *P* на поверхности объекта, и проекцией *P''* этой же точки *P*, построенной после устранения дисторсии с использованием параметров камеры.

Также, необходимо получить матрицу внешних параметров – это матрица поворота и смещения, переводящая координаты точек мировой сцены в систему координат, связанную с камерой. Так как камеры три, будет происходить калибровка системы камер, одна из которых будет полагаться началом координат, а остальные иметь поворот и смещение относительно первой. Ниже представлено уравнение перевода точки из мировой системы координат в систему координат камеры с учетом внешних параметров камеры:

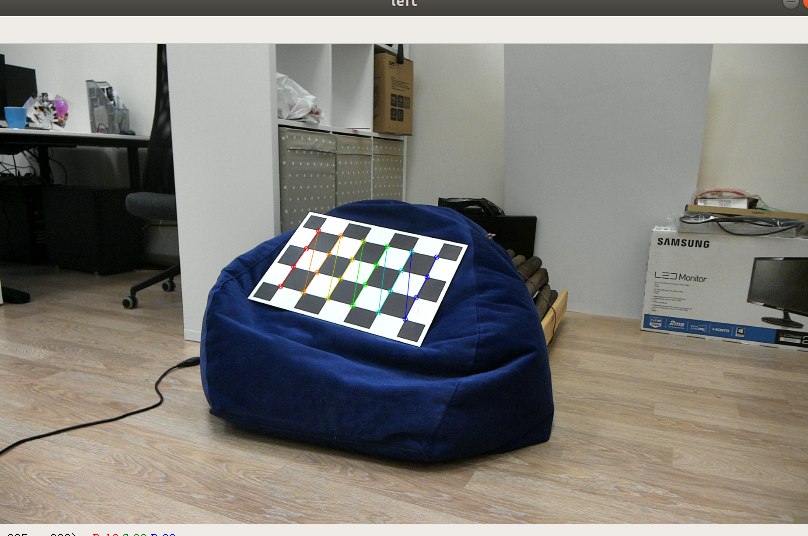
*,* u = , v = , где P – матрица проекции размером 3х4, состоящая из двух частей: внутренней матрицы , которая содержит внутренние параметры и внешней матрицы , которая является комбинацией матрицы вращения размером и вектора смещения .

Перечислим основные методы калибровки камеры [7]:

1. **Калибровка по шаблону:** когда процесс формирования изображений полностью контролируется, лучший способ выполнить калибровку — это сделать несколько изображений объекта или шаблона известных размеров с разных точек зрения.
2. **Геометрические примитивы**. Иногда в сцене присутствуют другие геометрические примитивы, например прямые линии и точки пересечений, которые также можно использовать для калибровки.
3. **На основе глубокого машинного обучения:** В случае ограниченности возможности управления настройками изображения (например, есть единственное изображение сцены), всё еще возможно получить информацию для калибровки камеры, используя Метод глубокого машинного обучения.

В данной работе будет использоваться калибровка по шаблону, так как есть возможность снять больше одной фотографии с калибруемых камер.

Для запуска алгоритма калибровки потребуется одновременный захват с трех камер шаблона с помощью функции *Opencv::VideoCapture()*, у которого известны координаты в 3D пространстве. В качестве такого шаблона возьмем шахматную доску с известной длиной клетки. Сделав порядка 15 фотографий шахматной доски с трех камер, можно получить внутренние и внешние параметры всех камер. Пример захвата изображения шахматной доски с камеры, с определенной позицией уголков представлен на Рис.3., Рис.4.



**Рис.3.** Пример захвата калибровочного паттерна pinhole камерой с определенными позициями уголков.

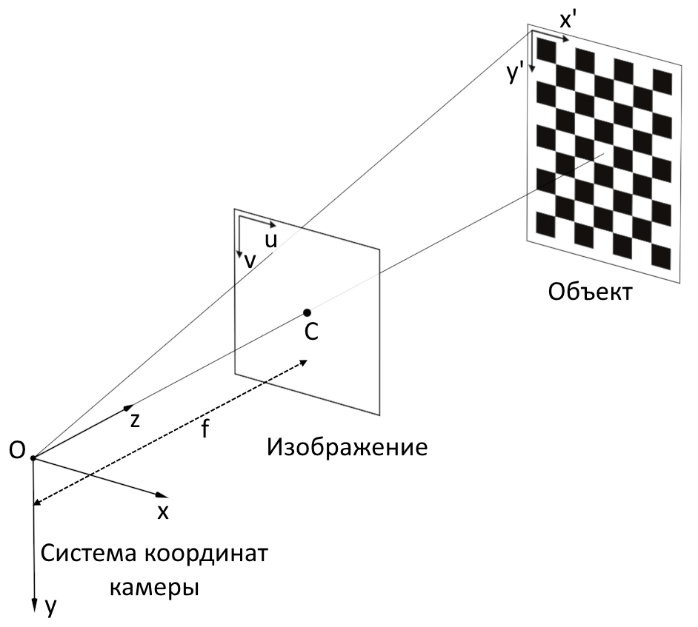


**Рис.4.** Пример захвата калибровочного паттерна pinhole камерой с определенными позициями уголков.

Алгоритм калибровки камеры включает в себя следующие шаги (с использованием библиотеки OpenCV):

1. Захват порядка 15 изображений одновременно с трех камер.
2. Нахождение уголков шахматной доски с помощью функции из *OpenCv::findChessboardCorners()*
3. Зная расстояние между уголками на фотографии и расстояние между уголками реальном мире (длина шахматной клетки = 7 см), можно получить внутренние и внешние параметры камеры, соотнося данные расстояния. Делается это с помощью функции *Opencv::*[*calibrateCamera*](https://docs.opencv.org/master/d9/d0c/group__calib3d.html#ga3207604e4b1a1758aa66acb6ed5aa65d)*().*
4. После этого, получив для каждой камеры внешние параметры - а именно, матрицу поворота и смещения мировых координат относительно системы координат камеры, простым перемножением матриц можно получить матрицу поворота и смещения для камер относительно друг друга.

Процесс калибровки камеры представлен на Рис.5.



**Рис.5.** Процесс калибровки камеры.

* + 1. **Построение облака точек реконструкции.**

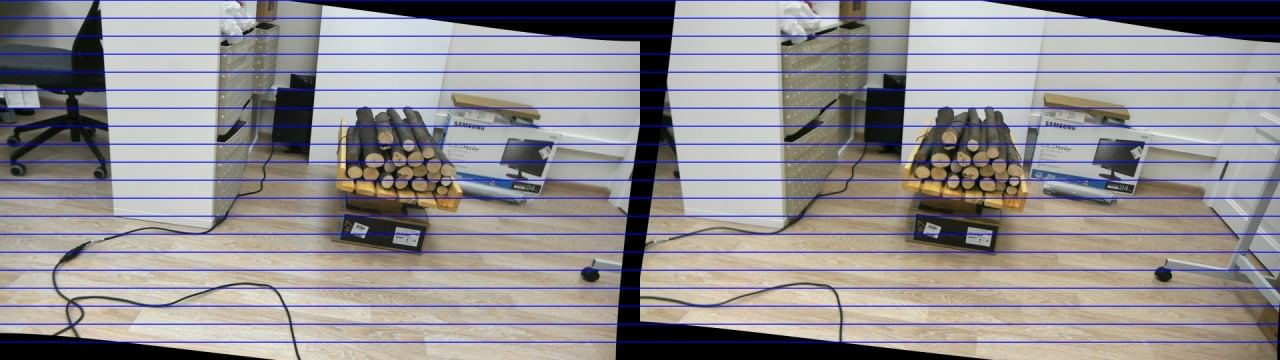
Зная внешние и внутренние параметры всех трех камер, можно произвести реконструкцию точек в 3D пространстве. В данном случае использовался алгоритм Stereo Block Matching. Данный алгоритм используется для двух камер для создания стереопары. Поэтому его необходимо использовать 3 раза для каждой пары камер: центральная, правая, центральная левая, левая правая. Прежде чем приступать к алгоритму Stereo BM предлагается рассмотреть понятие эпиполярной прямой. [8]



**Рис.6.** Пример построения эпиполярных прямых двух камер.

Пусть имеются две камеры, как изображено на Рис.6. — центр первой камеры,  — центр второй камеры. Точка пространства  проецируется в  на плоскость изображения левой камеры и в на плоскость изображения правой камеры. Прообразом точки  на изображении левой камеры является луч . Этот луч проецируется на плоскость второй камеры в прямую , называемую эпиполярной прямой. Образ точки  на плоскости изображения второй камеры обязательно лежит на эпиполярной прямой . Рассмотрение переходит к алгоритму Stereo Block Matching.[2],[3],[4].

1. Сначала два изображения выравниваются так, чтобы все эпиполярные линии были параллельны сторонам изображений, а также, чтобы для точки соответствующая ей эпиполярная линия задавалась уравнением . Таким образом, точке на одном изображении, нужно искать парную точку в той же строчке на втором изображении. Такой процесс называется ректификацией изображений. Пример ректифицированных изображений на Рис.7.



**Рис.7.** Пример ректифицированных изображений.

1. После того как изображения ректифицированы, выполняется поиск соответствующих пар точек. Для каждого пикселя левой картинки с координатами выполняется поиск пикселя на правой картинке. При этом предполагается, что пиксель на правой картинке должен иметь координаты где *d* — величина, называемая несоответствие/смещение (disparity). Поиск соответствующего пикселя выполняется путем вычисления максимума функции отклика, в качестве которой может выступать, например, корреляция окрестностей пикселей. В результате получается карта смещений (disparity map). Карта диспаритетов – матрица изображения, размеры которого совпадают с размерами исходного изображения, но при этом в каждом значении хранится смещение между двумя фотографиями по оси x. Пример disparity map изображен на Рис.8.



**Рис.8.** Пример карты диспаритетов.

1. Собственно, значения глубины обратно пропорциональны величине смещения пикселей. Зависимость между disparity и глубиной можно выразить следующим способом:

где, – значение глубины, – фокусное расстояние камеры, – смещение, а – расстояние между камерами. Таким образом, зная координаты и получив глубину точки можно получить трехмерные точки для стереореконструкции. Пример облака точек представлен на Рис.9.



**Рис.9.** Пример полученного облака точек.

В итоге, для каждой стереопары получается три облака точек, для левой и центральной камеры, для правой и центральной камеры, а также для левой и правой камер. Для того, чтобы получить одно облако в единой системе координат, выбирается камера, в систему координат которой необходимо привести оставшиеся два облака. Зная матрицу поворота и смещения камер относительной друг друга, необходимо лишь перемножить данные матрицы.

* 1. **Применение Нейросети для определения эллипсов торцов.**

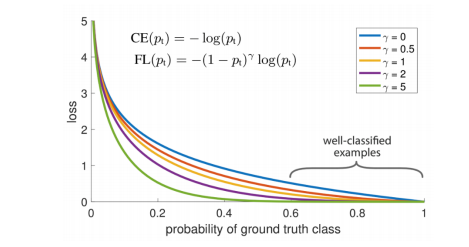
В качестве нейронной сети в данной работе используется RetinaNet [9],[10],[12]. RetinaNet – это одноступенчатый детектор, состоящий из базовой сети и двух специфичных для конкретных задач подсетей. Базовая сеть отвечает за вычисление карт признаков по всему входному изображению и представляет собой автономную сверточную сеть. Первая подсеть выполняет классификацию на выходе базовой сети, вторая подсеть выполняет поиск ограничивающей рамки. В качестве функции потерь RetinaNet использует Focal loss. Также в ней используется такой подход, как Featurized Image Pyramid.

**Функция потерь Focal loss**

Focal loss – это одна из модификаций cross entropy loss. Отличие заключается в добавлении параметра γ, который решает проблему несбалансированности классов. Во время обучения, большая часть объектов, обрабатываемых классификатором, является фоном, который является отдельным классом. Поэтому может возникнуть проблема, когда нейросеть обучится определять фон лучше, чем другие объекты. Добавление нового параметра решает данную проблему, уменьшив значение ошибки для легко классифицируемых объектов. Focal loss понижает вклад в обучение случаев, содержащих в себе только фон (easy negative) и наоборот фокусируется на случаях присутствия объекта или его маленькой части (hard negative).

Где – балансирующий параметр, который уравновешивает вклад положительных и отрицательных примеров.

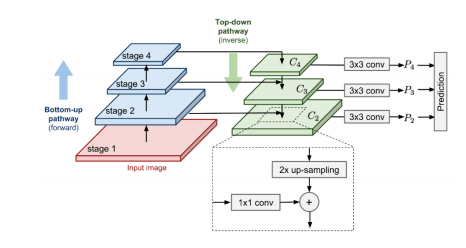
– фокусирующий параметр, который уравновешивает вклад easy negative и hard negative. График Focal loss при различных значениях представлен на Рис.10.



**Рис.10.** График Focal Loss при различных значениях .

**Функциональная пирамидальная сеть Featurized Pyramid Network**

FPN– это пирамидальная структура для извлечения карт признаков разных масштабов, полученных при сжатии размера входного изображения для дальнейшего их объединения с целью детектирования больших и малых объектов. Структура Featurized Image Pyramid изображена на Рис.11.



**Рис.11.** Структура Featuried Image Pyramid.

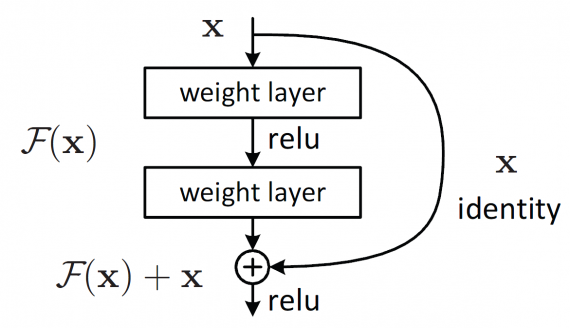
Feature Pyramid Network состоит из трёх основных частей: восходящий путь (bottom-up pathway), нисходящий путь (top-down pathway) и боковые соединения (lateral connections).

Восходящий путь реализует ResNet. Он состоит из нескольких модулей свертки, каждый из которых имеет много слоев свертки. По мере продвижения вверх пространственное измерение уменьшается вдвое (то есть удваивается шаг). Верхние слои сверточной сети имеют большее семантическое значение, но меньшее разрешение, а нижние наоборот. Восходящий путь имеет уязвимость при извлечении признаков – потеря важной информации об объекте, например из-за зашумления небольшого, но значимого, объекта фоном, так как к концу сети информация сильно сжата и обобщена. Выходные данные каждого модуля свертки помечаются и затем используются в нисходящем пути. При нисходящем пути, расширяя карту признаков предыдущего слоя на 2, используется метод ближайших соседей. Далее происходит поэлементное соединение расширенного слоя из нисходящего пути с соответствующим слоем из восходящего пути, к которому применяется свертка 1×1 для уменьшения размерности каналов. Наконец, к каждой объединенной карте применяется свертка 3×3 для создания финальной карты признаков, которая должна уменьшить эффект сглаживания при расширении.

Боковые соединения решают проблему затухания важных сигналов в процессе прохода по слоям, совмещая семантически важную информацию, полученную к концу первой пирамиды и более детальную информацию, полученную в ней ранее.

**ResNet**

В качестве базовой сети используется архитектура Resnet[9],[10],[12], которая решает задачу затухания градиента. Ее идея состоит в том, что чем глубже нейронная сеть, тем сложнее ее тренировать. Сложность тренировки заключается в большом количестве слоев, а при обратном распространении ошибки появляются проблемы с затуханием градиента. Ключевая особенность Resnet заключается в так называемых «**Соединения быстрого доступа» (shortcut connections). Данные соединения пропускают один или несколько слоев, как показано на Рис.12.**

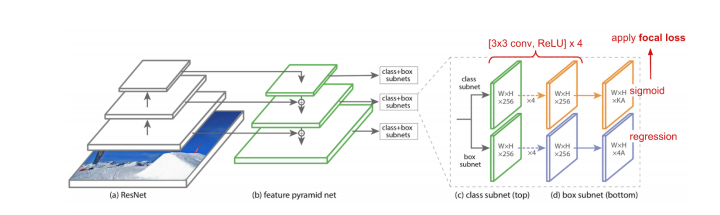


**Рис.12.** Соединения быстрого доступа, используемые в Resnet.

Выходы данных соединений добавляются к выходам из пропущенных слоев. После этого к ним применяется функция активации Relu. Таким образом, слои предсказывают не исходную функцию, а остаточную *.* А предсказывать остаточную функцию намного легче, чем изначальную сложную зависимость.Таким образом, можно получить более глубокую модель, которая не хуже менее глубокой, добавив несколько identity layers, то есть уровней, которые просто пропускают сигнал дальше без изменений. Дотренировать обычным способом до такой точности глубокие модели не получается. Поэтому у Resnet с увеличением количества слоев ошибка будет уменьшаться. Существует несколько разновидностей Resnet: Resnet-18, Resnet-34, Resnet-50, Resnet-101, Resnet-152. Цифры после названия указывают количество слоев в архитектуре. Resnet показывает самые лучшие результаты классификации на датасете ImageNet по сравнению со всем предыдущими нейронными сетями, а именно ResNet-152 показало 4.49% ошибок в top-5 - 5 самых подходящих классов к картинке и 19.38 в top-1.

Благодаря описанным особенностям данной нейронной сети, было решено использовать Resnet101.

Итоговая архитектура детектора RetinaNet с базовой сетью ResNet представлена на Рис.13.



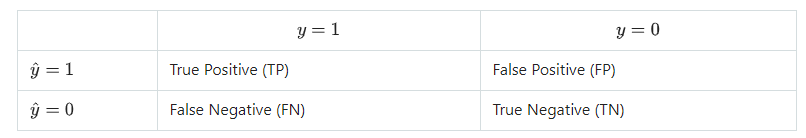
**Рис.13.** Итоговая архитектура детектора RetinaNet.

В данной конкретной задаче на вход нейросети RetinaNet подается изображение с той камеры, в систему координат которой было приведено три облака точек реконструкции. Выходом данной нейросети будут координаты центра, угол и оси эллипса для каждого из торцов. Пример работы сети RetinaNet представлен на Рис.14.



**Рис.14.** Пример работы нейросети RetinaNet на торцах бревен.

Как можно видеть на примере, Resnet недостаточно обучена, так как на изображении присутствуют множество false positive торцов. False positive – ошибка классификации, которую можно понять из матрицы ошибок на Рис.15. В данном случае, это означает, что нейросеть нашла торец бревна там, где его нет.



**Рис.15.** Матрица ошибок классификации нейросетей.

Данная проблема происходит из-за того, что обучающий датасет достаточно мал. Но при наличии false positive торцов, их можно будет достаточно легко отсечь, так как внутри данного эллипса будет малое количество (или вовсе не будет) точек 3d реконструкции.

* 1. **Алгоритм нахождения размеров торцов по нейронной сети и стерео-реконструкции.**

Переход к рассмотрению алгоритма, соединяющего построенную реконструкцию алгоритмами стереозрения и нейросеть RetinaNet.

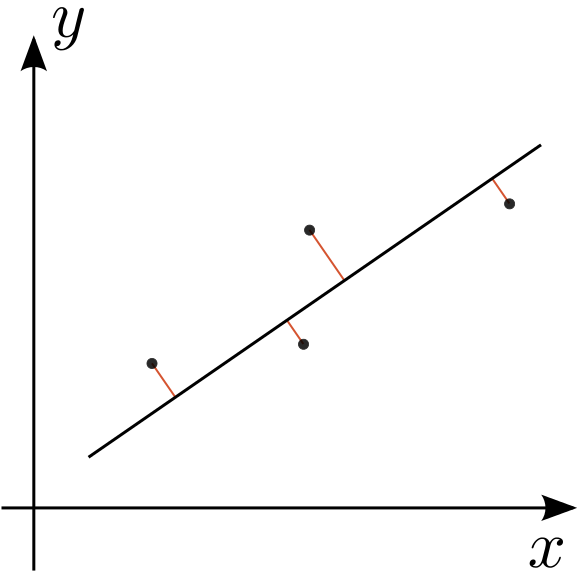
1. По выходу нейросети необходимо найти точки на границе эллипса. Для более точного построения эллипса выбирается не менее пятидесяти точек на эллипсе.
2. Каждую точку на границе эллипса необходимо преобразовать в 3D точку в пространстве, с помощью умножения на обратную матрицу камеры.
3. Пусть точка (0,0,0) является центром камеры, тогда соединяя точку (0,0,0) с проекцией точки на границе эллипса в пространстве, получаются лучи, образующие коническую поверхность в пространстве облака точек. Зная хотя бы 6 точек (в данном случае их будет как минимум 50), лежащих на конусе, можно составить общее уравнение конической поверхности.
4. Далее происходит проверка на то, какие точки 3d реконструкции лежат внутри конической поверхности, а какие вне. Для начала необходимо определить какой знак уравнения конической поверхности соответствует внутренней части, а какой внешней. Для этого берется точка, которая точно лежит внутри коники. Например, проекция точки центра эллипса. Затем отбрасываются те точки, подставление которых в уравнение коники дает знак, отличный от знака спроецированного центра эллипса.
5. Берутся только те точки, которые лежат внутри конической поверхности, которая соответствует одному торцу бревна.
6. Строится доминантная плоскость по найденным точкам внутри конической поверхности, используя алгоритм Lo-Ransac [14]. Прежде чем переходить к рассмотрению алгоритма Lo-Ransac, рассматривается, что такое SVD разложение матрицы или сингулярное разложение матрицы [15]. Сингулярное разложение матрицы порядка является разложение следующего вида:

Где - матрица размера с неотрицательными элементами, у которой значения, лежащие на главной диагонали – это сингулярные числа, а матрицы (порядка ) и (порядка ) – унитарные матрицы, состоящие из левых и правых сингулярных векторов соответственно. Свойствами SVD разложения можно назвать способность показывать [ранг матрицы](http://www.machinelearning.ru/wiki/index.php?title=%D0%A0%D0%B0%D0%BD%D0%B3_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D1%8B&action=edit), приближать матрицы данного ранга. Сингулярное разложение можно переформулировать в геометрических терминах. Линейный оператор, отображающий элементы пространства {\displaystyle \mathbb {R} ^{n}} в себя, представим в виде последовательно выполняемых линейных операторов вращения и растяжения. Теперь перейдем к рассмотрению алгоритма Lo-Ransac:

1. Из точек, лежащих внутри конуса, берется 3 случайные точки.
2. Производится оценка полученной плоскости, которая построена по трем выбранным точкам. Оценка производится с помощью SVD разложения матрицы, составленной из точек, полученных на первом шаге.
3. Считается количество точек, которое подходят для данной плоскости - инлаеров для каждой точки внутри конуса. Инлаерами считаются те точки, у которых расстояние до полученной плоскости меньше заранее заданного порога.
4. Если количество инлаеров превосходит заданный порог (в данном

случае это 70% от всех точек), то алгоритм переходит к шагу локальнойоптимизации

Прежде чем переходить к первому шагу локальной оптимизации рассмотрим, что такое ортогональная регрессия [16]. Ортогональная регрессия служит для вычисления наилучшего соответствия линии для конечного количества метрических масштабными пар данных по методу наименьших квадратов. Как и в других регрессионных моделях, сумма квадратов расстояний от прямой минимизирована. В отличие от других форм линейной регрессии, ортогональная регрессия использует не расстояния или направления, а ортогональные расстояния. Эта процедура не делает различий между независимой и зависимой переменной. Таким образом, в отличие от линейной регрессии, можно обрабатывать приложения, в которых обе переменные подвержены ошибкам измерения. Пример ортогональной регрессии представлен на Рис.16.



**Рис.16.** Пример ортогональной регрессии.

**Локальная оптимизация**

1. По всем точкам, вошедшим в число инлаеров, строится плоскость, методом ортогональной регрессии.
2. Если оценка данной плоскости лучше, чем оценка, полученная на предыдущем шаге, то данная плоскость становится оптимальной.

В алгоритме Lo-Ransac можно регулировать число попыток построения плоскости, а также порог, при котором точка определяется инлаером. Таким образом, можно достичь баланса между быстротой и точностью работы Lo-Ransac.

1. Находятся точки пересечения полученной доминантной плоскости с лучами, образующими коническую поверхность. Таким образом получается наиболее точное расположение плоскости торца бревна с учетом его “выдвинутости” относительно всего штабеля в мировой системе координат.
2. Точки пересечения переводятся в систему координат получившейся плоскости. Алгоритм перевода системы координат в систему координат плоскости:

Необходимо построить базис плоскости и определить смещение. Базис плоскости - упорядоченная пара линейно независимых векторов.

1. Смещением будут координаты точки пересечения плоскости и луча конуса из 6. (выбираем любую), эта точка будет началом координат в новой системе координат.
2. Первый вектор получается из нормали построенной плоскости.
3. Второй вектор получается из двух точек, лежащих на плоскости из п.6 (в данном случае их 50, выбираем 2 любые точки).
4. Третий вектор получается векторным произведением первых двух векторов.

5) Тогда матрица поворота и смещение

1. Новый эллипс вписывается по точкам, полученным при пересечении плоскости торца и конуса в новой системе координат.

Шаги 1-8 повторяются для каждого торца из штабеля. На выходе получается csv-файл, который содержит большой и малый диаметр эллипсов, которые будут соответствовать реальным размерам торцов.

* 1. **Улучшения**

**Фильтрация облаков**

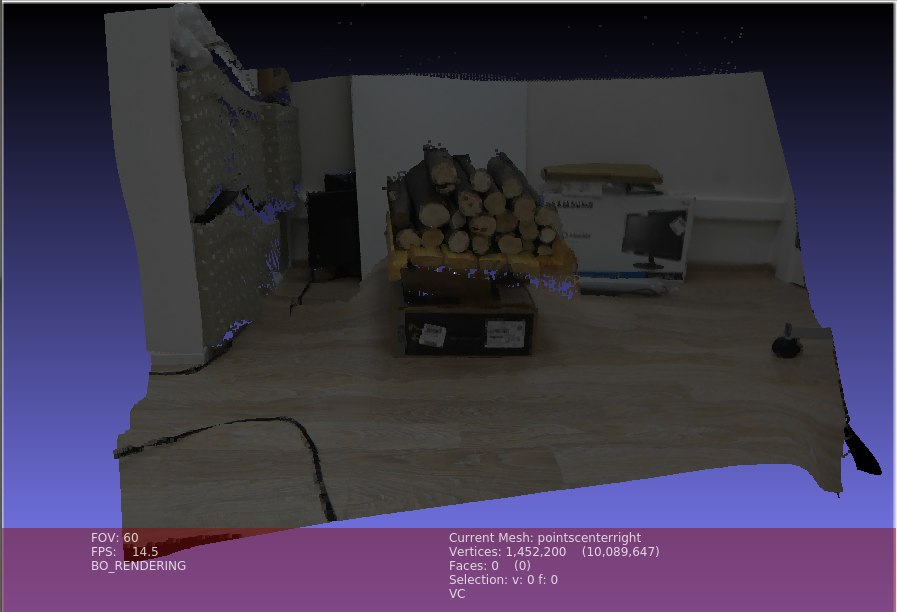
Для устранения выбросов на облаке точек используется сравнение карт диспаритетов, а именно: облака точек поворачиваются и смещаются так, чтобы быть в системе координат одной камеры. Соответствующие карты диспаритетов трех облаков сравниваются поточечно следующим образом:

1. Если точка присутствует как минимум на двух облаках, то данная точка остается в рассмотрении
2. Если точка присутствует только на одном из трех облаков, убираем точку из рассмотрения.

Данный метод помогает отфильтровать итоговое облако точек от шума.

**Сглаживание облаков**

Применение сглаживающего фильтра GaussianBlur из OpenCV [17]. GaussianBlur выполняется путем свертки каждой точки входного массива гауссовым ядром, а затем суммирования их всех для получения выходного массива. Данный метод не помог в улучшении точности определения размеров торцов, но облака точек стали более объемными. Пример работы сглаживающего фильтра представлен на Рис.17.



**Рис.17.** Пример работы сглаживающего фильтра.

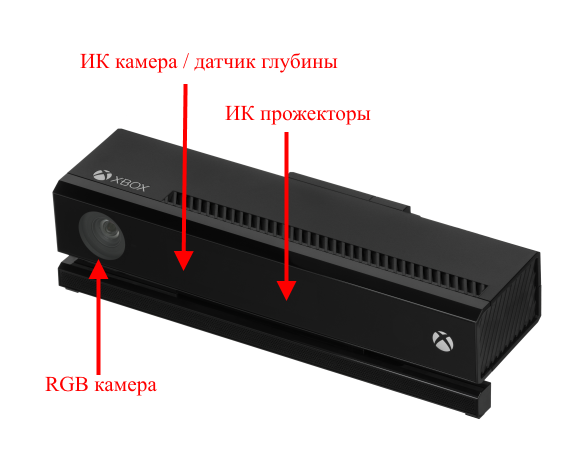
**Глава 2. Алгоритм нахождения объема штабеля лесовоза по движущемуся лесовозу.**

Теперь перейдем к рассмотрению второго алгоритма, который будет вычислять объем всего штабеля по движущемуся лесовозу

Данный алгоритм также состоит из нескольких этапов:

1. Захват изображений с камер
2. Построение плотной реконструкции по видео/серии фото
3. Определение габаритов штабеля по 3d облаку точек.
   1. **Захват изображений с камер**

В данном алгоритме используются не три, а шесть камер. Пять из них, обычные pinhole камеры, а оставшаяся - rgbd камера Microsoft Kinect v2.0. Rgbd камера может снимать не только rgb-изображение, но и инфракрасные изображения, а также автоматически строить карту глубин, тогда как в прошлом алгоритме для этого требовались стереопары и алгоритм Stereo Block Matching. Расположение датчиков Rgbd камеры представлен на Рис.18. [11]



**Рис.18.** Расположение датчиков камеры Microsoft Kinect v2.0.

Для данного алгоритма рамку необходимо модифицировать следующим образом: изменить ее длину с одного метра до двух. Это сделано для того, чтобы боковые части проезжающего лесовоза полностью помещались в угол обзора боковых камер. Расположение камер будет следующим – по две камеры необходимо расположить с левой, верхней и правой стороны. Rgbd камеру можно расположить с любой из сторон. Пример расположения камер и рамки изображен на Рис. 19. В данном случае производится проезд тележки с бревнами, а камеры производят одновременный захват раз в секунду.



**Рис.19.** Пример расположения пяти pinhole камер и одной rgbd-камеры на металлической рамке.

* 1. **Построение плотной реконструкции по серии фото.**

В качестве построение плотной реконструкции используется Colmap [18] – это пайплайн для Structure from motion. Алгоритм работы Colmap:

1. На каждом изображении используется алгоритм scale-invariant feature transform (sift) [13] - является алгоритмом выявления и описания локальных признаков в изображениях. Состоит данный алгоритм из следующих шагов:
   1. **Нахождение особых точек.**

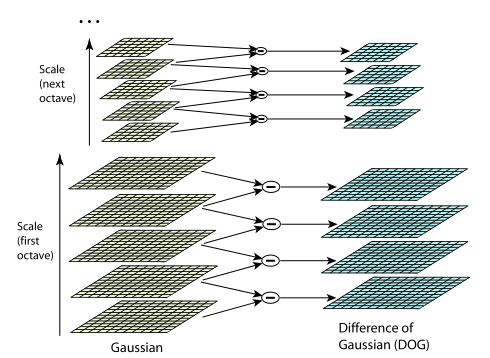
Основным моментом в детектировании особых точек является построение пирамиды гауссианов и разностей гауссианов. Гауссианом (или изображением, размытым гауссовым фильтром) является изображение:

Где – значение Гауссиана в точке с координатами , а – радиус размытия. -гауссово ядро, – значение исходного изображения, - операция свертки.

Разностью гауссианов называют изображение, полученное путем попиксельного вычитания одного гауссина исходного изображения из гауссиана с другим радиусом размытия.

Различная степень размытия изображения гауссовым фильтром может быть принята за исходное изображение, взятое в некотором масштабе.

Инвариантность относительно масштаба достигается за счет нахождения ключевых точек для исходного изображения, взятого в разных масштабах. Для этого строится пирамида гауссианов: все масштабируемое пространство разбивается на некоторые участки — октавы, причем часть масштабируемого пространства, занимаемого следующей октавой, в два раза больше части, занимаемой предыдущей. К тому же, при переходе от одной октавы к другой делается ресэмплинг изображения, то есть его размеры уменьшаются вдвое. Естественно, что каждая октава охватывает бесконечное множество гауссианов изображения, поэтому строится только некоторое их количество , с определенным шагом по радиусу размытия. С тем же шагом достраиваются два дополнительных гауссиана (всего получается ), выходящие за пределы октавы. Масштаб первого изображения следующей октавы равен масштабу изображения из предыдущей октавы с номером . Параллельно с построением пирамиды гауссианов, строится пирамида разностей гауссианов, состоящая из разностей соседних изображений в пирамиде гауссианов. Соответственно, количество изображений в этой пирамиде будет . На Рис.20. слева изображена пирамиды Гауссианов, а справа их разности.



**Рис.20.** Пирамида Гауссианов и их разностей.

 Точка считается особой, если она является локальным экстремумом разности гауссианов. То есть если эта точка больше (меньше) всех соседей, то она принимается за точку локального экстремума.

* 1. **Уточнение особых точек**

 Следующим шагом будет проверка пригодности точки экстремума на роль ключевой.  
Первым делом определяются координаты особой точки с субпиксельной точностью. Это достигается с помощью аппроксимирования функции разности гауссианов многочленом Тейлора второго порядка, взятого в точке вычисленного экстремума.

Где -функция разностей гауссианов, x – вектор смещения относительно точки разложения. Экстремум многочлена Тейлора находится путем вычисления производной и приравнивания ее к нулю. В итоге получается смещение точки вычисленного экстремума, относительно точного.

Если одна из компонент вектора больше половина шага сетки в этом направлении, то это означает, что на самом деле точка экстремума была вычислена неверно и нужно сдвинуться к соседней точке в направлении указанных компонент. Для соседней точки все повторяется заново. Если таким образом алгоритм выходит за пределы октавы, то следует исключить данную точку из рассмотрения.

Когда положение точки экстремума вычислено, проверяется на малость само значение DoG в этой точке по формуле:

Если эта проверка не проходит, то точка исключается, как точка с малым контрастом.

Вторая проверка.  Если особая точка лежит на границе какого-то объекта или плохо освещена, то такую точку можно исключить из рассмотрения. Эти точки имеют большой изгиб (одна из компонент второй производной) вдоль границы и малый в перпендикулярном направлении. Этот большой изгиб определяется матрицей Гессе . Для проверки подойдет размера 2x2.

Пусть — след матрицы, а — её определитель.

Пусть r — отношение большего изгиба к меньшему,

Тогда

Точка рассматривается дальше, если

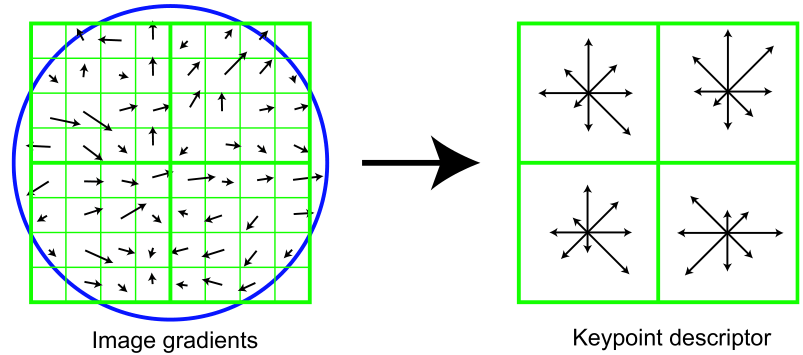
* 1. **Нахождение ориентации ключевой точки.**

После того, как стало известно, что какая-то точка является ключевой, нужно вычислить её ориентацию.   
Направление ключевой точки вычисляется исходя из направлений градиентов точек, соседних с особой. Все вычисления градиентов производятся на изображении в пирамиде гауссианов, с масштабом наиболее близким к масштабу ключевой точки. Величина и направление градиента в точке вычисляются по формулам:

Определяется окно (окрестность) ключевой точки, в котором будут рассмотрены градиенты. По сути, это будет окно, требуемое для свертки с гауссовым ядром, причем оно будет круглым и радиус размытия для этого ядра равен 1.5 масштаба ключевой точки. Для гауссова ядра действует так называемое правило «трех сигм». Оно состоит в том, что значение гауссова ядра очень близко к нулю на расстоянии, превышающем 3. Таким образом, радиус окна определяется как 3.  
Направление ключевой точки находится из гистограммы направлений O. Гистограмма состоит из 36 компонент, которые равномерно покрывают промежуток в 360 градусов, и формируется она следующим образом: каждая точка окна вносит вклад, равный , в ту компоненту гистограммы, которая покрывает промежуток, содержащий направление градиента .  
Направление ключевой точки лежит в промежутке, покрываемом максимальной компонентой гистограммы. Значения максимальной компоненты и двух соседних с ней интерполируются параболой, и точка максимума этой параболы берётся в качестве направления ключевой точки. Если в гистограмме есть ещё компоненты с величинами не меньше , то они аналогично интерполируются и дополнительные направления приписываются ключевой точке

* 1. **Нахождение дескрипторов**

Теперь перейдем непосредственно к дескрипторам. В принципе, дескриптором может выступать любой объект (лишь бы он справлялся со своими функциями), но обычно дескриптором является некая информация об окрестности ключевой точки. Такой выбор сделан в силу нескольких причин: на маленькие области меньшее влияние оказывают эффекты искажений, некоторые изменения (изменение положения объекта на картинке, изменение сцены, перекрытие одного объекта другим, поворот) могут не повлиять на дескриптор вовсе.  
  
В методе SIFT дескриптором является вектор. Как и направление ключевой точки, дескриптор вычисляется на гауссиане, ближайшем по масштабу к ключевой точке, и исходя из градиентов в некотором окне ключевой точки. Перед вычислением дескриптора это окно поворачивается на угол направления ключевой точки, чем и достигается инвариантность относительно поворота. Изображение и полученный для него дескриптор изображены на Рис.21.



**Рис.21**.Изображение и полученный на его основе дескриптор.

Здесь схематично показана часть изображения и полученный на её основе дескриптор. В левой части Рис.21. можно видеть пиксели, обозначенные маленькими квадратиками. Эти пиксели берутся из квадратного окна дескриптора, которое в свою очередь поделено ещё на четыре равных части (дальше будут называться их регионами). Маленькая стрелочка, в центре каждого пикселя обозначает градиент этого пикселя. Интересно то, что центр этого окна находится между пикселями. Его надо выбирать как можно ближе к точным координатам ключевой точки. Последняя деталь, которую можно увидеть — это круг, обозначающий окно свертки с гауссовым ядром (аналогично окну для вычисления направления ключевой точки). Для этого ядра определяется 𝞂, равное половине ширины окна дескриптора. В дальнейшем значение каждой точки окна дескриптора будет домножаться на значение гауссова ядра в этой точке, как на весовой коэффициент.  
  
В правой части Рис.21. можно видеть схематически изображенный дескриптор особой точки, размерности . Первые две цифры в значении размерности — это количество регионов по горизонтали и вертикали. Те квадраты, которые охватывали некоторый регион пикселей на левом изображений, справа охватывают гистограммы, построенные на пикселях этих регионов. Соответственно, третья цифра в размерности дескриптора означает количество компонент гистограммы этих регионов. Гистограммы в регионах вычисляются так же, как и гистограмма направлений с тремя небольшими, но:  
1. Каждая гистограмма так же покрывает участок в 360 градусов, но делит его на 8 частей.  
2. В качестве весового коэффициента берется значение гауссова ядра, общего для всего дескриптора.  
3. В качестве ещё одних весовых коэффициентов берутся коэффициенты трилинейной интерполяции.  
  
Каждому градиенту в окне дескриптора можно приписать три вещественные координаты , где — расстояние до градиента по горизонтали, — расстояние по вертикали, — расстояние до направления градиента в гистограмме (имеется ввиду соответствующая гистограмма дескриптора, в которую вносит вклад этот градиент). За точку отсчета принимается левый нижний угол окна дескриптора и начальное значение гистограммы. За единичные отрезки берутся размеры регионов по горизонтали и вертикали для и соответственно, и количество градусов в компоненте гистограммы для n. Коэффициент трилинейной интерполяции определяется для каждой координаты градиента как , где равно расстоянию от координаты градиента до середины того единичного промежутка, в который эта координата попала. Каждое вхождение градиента в гистограмму умножается на все три весовых коэффициента трилинейной интерполяции.

Дескриптор ключевой точки состоит из всех полученных гистограмм. Размерность дескриптора на 32 компоненты но на практике используются дескрипторы размерности 128 компонент .  
  
Полученный дескриптор нормализуется, после чего все его компоненты, значение которых больше 0.2, урезаются до значения 0.2 и затем дескриптор нормализуется ещё раз. В таком виде дескрипторы готовы к использованию.

Поиск дескрипторов происходит для каждого изображения из серии захваченных фото в процессе проезда лесовоза через металлическую рамку.

1. Далее происходит матчинг между изображениями, то есть изображения с каждой камеры сравниваются и ищутся одинаковые дескрипторы.
2. Таким образом, появляется разряженное облако точек. Пример разряженного облака представлен на Рис.22. Каждая из красных полос это движение определенной камеры. Как можно увидеть, все шесть камер распознались. Rgbd камера в данном случае выступает как обычная pinhole камера, используется только rgb изображение.



**Рис.22.** Пример построенной разряженной реконструкции штабеля.

1. Теперь необходимо из разряженной реконструкции получить плотную [18]. Получение плотной реконструкции достигается путем интерполяции разряженной реконструкции. Пример плотной реконструкции из получившегося разряженного облака представлен на Рис.23.



**Рис.23.** Пример плотного облака.

1. Чтобы восстановить масштаб штабеля необходима Rgbd камера. Так как данная камера может определять расстояние до объектов и строить карту глубин, то необходимо определить положение камеры до объекта, а значит определить внешние параметры камеры. Так как все камеры определились верно на разряженном облаке, то зная матрицу поворота и смещение одной из них, мы знаем внешние параметры для всех камер.
2. Далее применяется фильтрация облаков. Используются стандартные методы из библиотеки Open3d, которая создана для работы с 3d облаками. Данная функция применяется так, чтобы на 3d реконструкции остался только штабель. Например, в данном случае использовалась функция *remove\_statistical\_outlier()* [19], которая удаляет точки, которые находятся дальше от своих соседей по сравнению со средним значением для облака точек.
3. Для ограничения штабеля лесовоза параллелепипедом, используется функция из *OpenCV – findBoundingBox()* [20]. Данная функция строит такой параллелепипед, чтобы все точки в облаке были вписаны в него. На выходе получаются размеры параллелепипеда, в который описывает данный штабель.
   1. **Улучшения**

Для того, чтобы легче было отсекать штабель от фона, необходимо применить маски для каждого изображения. Для этого использовался алгоритм LSBP (Local SVD Binary Pattern) [21] - алгоритм вычитания фона из изображения с использованием локального бинарного паттерна SVD. Бинарный SVD паттерн хорошо находит локальные дескрипторы на изображении и при следующем сравнении двух изображений может распознать сходства и различия. Для его работы необходимо захватить изображения со всех камер без штабеля.

**Результаты**

Среднее отклонение действительных размеров торцов от выходов алгоритма разнится от фотографии к фотографии, но, в среднем, составляет от 5 до 10 процентов. Результаты представлены в Табл.1. Подход второго алгоритма показывает расхождение на 1, 65% в среднем по параметрам высоты, ширины и длины тележки с реальными результатами. Результаты второго подхода представлены в Табл.2

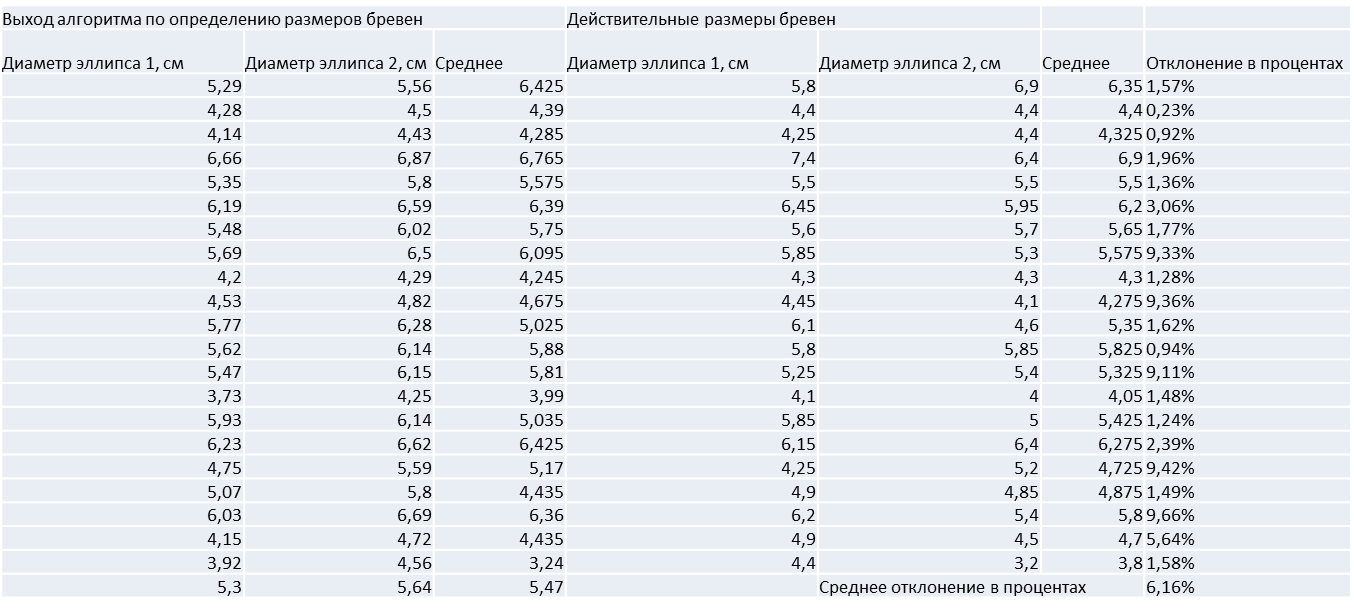
Табл.1. Результаты работы алгоритма по определению размеров торцов бревен.



Табл.2. Результаты работы алгоритма по определению объема штабеля движущегося лесовоза.

**Заключение**

Алгоритмы стереозрения являются очень важной темой для многих областей жизни. Нейросети также на данный момент используются во многих процессах повседневной жизни. Две данные технологии в совокупности дали решение, которое имеет значительный результат в поставленной задаче определения размеров торцов бревен, и показанное среднее отклонение результатов от действительных размеров показывает преимущество над человеческим показателем среднего отклонения. Был придуман и реализован алгоритм совмещения нейросетей и алгоритмов стереозрения. Проведена серия экспериментов, по результатам которой удалось с достаточной точностью определить размеры торцов бревен. Также, для второго алгоритма был найден подход совмещения алгоритмов Structure from motion с возможностями RGBd камеры. Были также проделаны эксперименты и получены результаты, точность которых превышает точность существующих методов на предприятиях с приемкой леса.

**Литература**

[1] Hartley R., Zisserman A. Multiple view geometry in computer vision. // Cambridge university press, 2003. 350c.

[2] Szeliski R. Computer vision: algorithms and applications // Springer Science & Business Media, 2010. 620c.

[3] Bradski G., Kaehler A. Learning Opencv // O’Reilly Media, 2008, 571c.

[4] Kramer J. Hacking the Kinect // Apress. 2012. 340c.

[5] Lachat E. et al. First Experiences with Kinect V2 Sensor for Close Range 3D Modelling // International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. 2015. 150 c.

[6] Gao X.S. et al. Complete solution classification for the perspective-three-point problem // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2003. 650-700 c.

[7] Калибровка Kinect v2 с помощью OpenCV [Электронный ресурс]: URL: <https://habr.com/ru/post/272629/> (дата обращения: 21.03.2021)

[8] Основы стереозрения [Электронный ресурс]:URL: <https://habr.com/ru/post/130300/> (дата обращения: 21.03.2021)

[9] Эволюция нейросетей для распознавания изображений в Google: Inception-Resnet [Электронный ресурс]: URL: <https://habr.com/ru/post/303196/> (дата обращения: 25.03.2021)

[10] Архитектура нейронной сети RetinaNet [Электронный ресурс]: URL: <https://habr.com/ru/post/510560/> (дата обращения 25.03.2021)

[11] Что нас ждет в Microsoft Kinect 2.0? [Электронный ресурс]: URL: <https://habr.com/ru/company/veeam/blog/196636/> (дата обращения: 15.04.2021)

[12] Inception-v4, Inception-Resnet and the Impact of Residual Connections on Learning [Электронный ресурс]: URL: <http://arxiv.org/abs/1602.07261> (дата обращения: 15.04.2021)

# [13] Построение SIFT дескрипторов и задача сопоставления изображений [Электронный ресурс]: URL: [https://habr.com/ru/post/106302/](https://habr.com/ru/post/106302/%20)

# (дата обращения 25.04.2021)

# [14] RANSAC [Электронный ресурс]: URL: <https://ru.m.wikipedia.org/wiki/RANSAC/> (дата обращения 25.04.2021)

# [15] Сингулярное разложение [Электронный ресурс]:URL: <https://ru.m.wikipedia.org/wiki/Сингулярное_разложение>/ (дата обращения 25.04.2021)

# [16] Регрессия Деминга [Электронный ресурс]: URL: <https://ru.m.wikipedia.org/wiki/Регрессия_Демпинга/> (дата обращения 25.04.2021)

# [17] Image Blurring [Электронный ресурс]: URL: <https://docs.opencv.org/master/d4/d13/tutorial_py_filtering.html> (дата обращения 25.04.2021)

# [18] COLMAP [Электронный ресурс]: URL: <https://colmap.github.io> (дата обращения 26.04.2021)

# [19] Point cloud outlier removal [Электронный ресурс]: URL: <https://www.open3d.org/docs/latest/tutorial/Advanced/pointcloud_outlier_removal.html> (дата обращения 26.04.2021)

# [20] Creating Bounding Boxes and circles for contours [Электронный ресурс]:URL: <https://docs.opencv.org/3.4/da/d0c/tutorial_bounding_rects_circles.html> (дата обращения 27.04.2021)

# [21] Guo L., Xu D., Zhenping Q., Background Subtraction using Local SVD Binary Pattern//Computer Vision Foundation, 2016, 9c.