МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ РАДИОФИЗИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ

Кафедра информатики и компьютерных систем

СРАВНЕНИЕ ВЫЧИСЛИТЕЛЬНОЙ ЭФФЕКТИВНОСТИ АЛГОРИТМОВ ПОИСКА ЦЕНТРА ПЯТНА ОСВЕЩЁННОСТИ КМОП-МАТРИЦЫ ДАТЧИКА НАПРАВЛЕНИЯ НА СОЛНЦЕ

Курсовая работа

Барковского Ярослава Юрьевича,
студента 4 курса
специальность «радиофизика»
Научный руководитель:
ассистент кафедры ИиКС РФиКТ
С.В. Василенко

Минск, 2020

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В современном мире Солнце является основным навигационным ориентиром, особенно для спутников, поэтому все спутники оснащаются приборами, получившими название солнечных датчиков(СД) или по-другому датчиками солнечной ориентации. Первые упоминания о приборах ориентации по Солнцу в истории космонавтики относятся к первым запускам искусственных спутников Земли. Это оптико-электронные приборы служили для поиска Солнца и формирования электрических сигналов, пропорциональных направлению на энергетический центр диска Солнца в связанной со спутником системе координат. Эти сигналы используются бортовой системой управления либо для разворота спутника в процессе обеспечения его требуемой угловой ориентации на Солнце (например, для ориентации жестко закрепленных на корпусе спутника солнечных батарей или антенны радиопередатчика), либо для последующего расчета места ориентации спутника в пространстве. В первом случае прибор называется датчиком угловой ориентации спутника, а во втором – датчиком углового положения Солнца. Оба типа приборов делятся по точностным характеристикам разделяются на грубые и точные датчики. Принято считать, что грубые — это датчики, с погрешностью более 5°, умеренной точности, с погрешностью от 0,5 до 5° и точные, с погрешностью менее 30 угл.мин.

Разработано множество вариантов схемотехнического исполнения СД, обеспечивающих как грубое, так и точное измерение направления на Солнце, использующих как статичные, так и подвижные составные элементы. Учитывая современные тенденции развития исследований Земли из космоса, включая задачи дистанционного зондирования Земли, стоит отметить, что все большее значение принимает концепция использования сверхмалых космических аппаратов таких как наноспутники формата CubeSat с линейными размерами порядка 10 см.

Среди датчиков определения положения солнечные датчики представляют собой простую и надежную технологию, используемую во многих космических полетах, позволяющую определять относительное положение тела относительно Солнца, измеряя угол падения солнечного излучения сквозь небольшое отверстие. Общая базовая линия для двухосных цифровых датчиков Солнца состоит в массиве активных пикселей, расположенных за небольшой апертурой: положение пятна, освещенного солнечными лучами, позволяет определить направление Солнца. С появлением более компактных транспортных средств, таких как CubeSat и наноспутники, возникла необходимость ограничить размер и вес таких устройств: в качестве компромисса это обычно приводит к значительному снижению производительности. В настоящее время стандартные готовые компоненты для CubeSat имеют точность до 0,3 ° с полем обзора от ± 45

° до ± 90 ° и стоят от нескольких тысяч долларов. В данной работе будет представлен недорогой миниатюрный датчик Солнца на основе коммерческой CMOS-камеры. В устройстве используется прецизионная диафрагма 20 мкм с зазором 7 мм с CMOS. Геометрия и конструкция обеспечивают максимальное разрешение менее 0,05 °, превосходя большинство доступных в настоящее время коммерческих решений. Природа технологии позволяет уменьшить размер, а также ограничить вес.

Задачей данной курсовой работы является сравнение вычислительной эффективности алгоритмов поиска центра изображения светового пятна на светочувствительной матрице CMOS кратко описанной выше. Для достижения данной промежуточно были сгенерированы изображения падающего луча на светочувствительную матрицу.

ГЛАВА 1
СВЕТОЧУВСТВИТЕЛЬНАЯ МАТРИЦА

Светочувствительная матрица (фотоматрица) – специализированная цифро-аналоговая или аналоговая интегральная микросхема, состоящая из светочувствительных элементов – фотодиодов. Основная задача фотоматриц это преобразование спроецированного на них оптического изображения в аналоговый электрический сигнал или в поток цифровых данных (если в матрице присутствует АЦП). Широкое применение получили в цифровых фотоаппаратах, видеокамерах и в системах солнечной и астронавигации.

1.1 Общие характеристики

Основными характеристиками фотоматриц являются: отношение сигнал/шум, чувствительность, разрешение, физический размер матрицы, коэффициент заполнения, отношение сторон кадра и пропорции пикселя. Рассмотрим каждую в отдельности:

- Отношение сигнал/шум в общем понятии - безмерная величина, которая отображает отношение между мощностью полезного сигнала и мощностью шума. В случае фотоматриц всякое физическое тело совершает некоторые колебания от своего среднего состояния, в науке это называется флуктуациями. Поэтому и каждое свойство всякого тела тоже изменяется, колеблясь в некоторых пределах. Это справедливо и для такого свойства, как светочувствительность фотоприемника, независимо от того, что собой представляет этот фотоприемник. Следствием этого является то, что некоторая величина не может иметь какого-то конкретного значения, а изменяется в зависимости от обстоятельств. Если, например, рассмотреть такой параметр фотоприемника, как «уровень чёрного», то есть то значение сигнала, которое будет показывать фотодатчик при отсутствии света, то и этот параметр будет каким-то образом флуктуировать.
- Чувствительность фотоматрицы характеризует степень ее реакции на условия окружающего освещения, то есть, чем меньшее количество световой энергии необходимо для получения изображения, тем выше светочувствительность матрицы. Самая частая причина получения изображений низкого качества – плохая освещенность объекта. Вообще, чем лучше освещенность, тем лучше изображение.
- Разрешение — способность оптического прибора воспроизводить изображение близко расположенных объектов. Разрешение матриц зависит от их типа, площади и плотности фоточувствительных элементов

на единицу поверхности. Оно нелинейно зависит от светочувствительности матрицы и от заданного программой уровня шума.

- У современных цифровых фотоматриц разрешающая способность определяется размером пикселя, который варьируется у разных фотоматриц в пределах от 0,0025 мм до 0,0080 мм, а у большинства современных фотоматриц он равен 0,006 мм. Поскольку две точки будут различаться, если между ними находится третья (незасвеченная) точка, то разрешающая способность соответствует расстоянию в два пикселя, то есть:

$$M = \frac{1}{2p}$$

где p – размер пикселя.

Физические размеры фотосенсоров определяются размером отдельных пикселей матрицы, которые в современных фотосенсорах имеют величину 0,005-0,006 мм. Чем крупнее пиксель, тем больше его площадь и количество собираемого им света, поэтому тем выше его светочувствительность и лучше отношение сигнал/шум.

- Коэффициент заполнения из датчика изображения массива представляет собой отношение светочувствительной области пикселя к его общей площади. Для пикселей без микролинз коэффициент заполнения представляет собой отношение площади фотодиода к общей площади пикселя, но использование микролинз увеличивает эффективный коэффициент заполнения, часто почти до 100%, путем схождения света от всей площади пикселя в фотодиод

Другим случаем, который уменьшает коэффициент заполнения изображения, является добавление дополнительной памяти рядом с каждым пикселем, чтобы добиться глобального эффекта затвора на датчике CMOS.

- Существует несколько основных отношений сторон кадра которые используются во всех матрицах:
    - Формат кадра 4:3
    - Формат кадра 3:2
    - Формат кадра 16:9
- Пропорции пикселя. Выпускаются светочувствительные матрицы с тремя различными пропорциями пикселя:
    - Для видеоаппаратуры выпускаются сенсоры с пропорцией пикселя 4:3
    - Так же некоторые выпускают светочувствительные матрицы с пропорцией пикселя 3:4

- Светочувствительные матрицы для астрономического, фотографического и рентгенографического оборудования обычно имеют квадратный пиксель, то есть пропорция сторон пикселя 1:1

Так же светочувствительные матрицы различаются методами получения цветных изображений, так как пиксели сами по себе являются “черно-белым”. Рассмотрим 3 основных системы: трехматричная система, матрицы с мозаичным фильтром и матрицы с полноцветными пикселями.

- В трехматричных системах свет попадает на дихроидные призмы делится на три основных цвета, после чего каждый цвет направляется на отдельную матрицу. Такие системы используются в видеокамерах высокого класса.
- В матрицах с мозаичными фильтрами пиксели расположены в одной плоскости и каждый пиксель накрыт светофильтром какого-то цвета. А недостающая информация в таких матрицах восстанавливается путем интерполяции. Основные типы расположения светофильтров это RGGB, RGBW, RGEB и CGMY.
- Существуют две технологии матриц с полноцветными пикселями, позволяющими получать с каждого пикселя все три цветовые координаты. Первая это многослойные матрицы используемы фирмой Foeveon, а вторая это полноцветная RGB-матрица от Nikon.
    - Фотоматрица от Foeveon называется Foeveon X3, где X3 означает трехслойность и трехмерность. В ней цветоделение на аддитивные цвета RGB проводится послойно, по толщине полупроводникового материала, с использованием физических свойств кремния.
    - В полноцветных матрицах Nikon лучи RGB предметных точек в каждом пикселе, содержащем одну микролинзу и три фотодиода, проходят через открытую микролинзу и падают на первое дихроичное зеркало. При этом синяя составляющая пропускается первым дихроичным зеркалом на детектор синего, а зелёная и красная составляющие отражаются на второе зеркало. Второе дихроичное зеркало отражает зелёную составляющую на детектор зелёного, и пропускает красную и инфракрасную составляющие. Третье дихроичное зеркало отражает красную составляющую на детектор и поглощает инфракрасную составляющую. Из-за сложностей в технологии этот патент не нашел достойного применения.

Светочувствительные матрицы различают по применяемой в них технологии. Существует два основных типа – ПЗС-матрицы и КМОП-матрицы.

1.2 ПЗС- матрицы

ПЗС-матрица – специализированная аналоговая интегральная микросхема, состоящая из светочувствительных фотодиодов, выполненная на основе кремния, использующая технологию ПЗС — приборов с зарядовой связью.

Прибор с зарядовой связью был изобретён в 1969 году Уиллардом Бойлом и Джорджем Смитом в Лабораториях Белла. Лаборатории работали над видеотелефонией и развитием «полупроводниковой пузырьковой памяти». Приборы с зарядовой связью начали свою жизнь как устройства памяти, в которых можно было только поместить заряд во входной регистр устройства. Однако способность элемента памяти устройства получить заряд благодаря фотоэлектрическому эффекту сделала данное применение ПЗС устройств основным. В 1970 году исследователи Bell Labs научились снимать изображения с помощью простых линейных устройств. Название ПЗС — прибор с зарядовой связью — отражает способ считывания электрического потенциала методом сдвига заряда от элемента к элементу. ПЗС устройство состоит из поликремния, отделённого от кремниевой подложки, у которой при подаче напряжения через поликремневые затворы изменяются электрические потенциалы вблизи электродов.

В основе работы ПЗС лежит явление внутреннего фотоэффекта. Один элемент ПЗС-матрицы формируется тремя или четырьмя электродами. Положительное напряжение на одном из электродов создаёт потенциальную яму, куда устремляются электроны из соседней зоны. Последовательное переключение напряжения на электродах перемещает потенциальную яму, а, следовательно, и находящиеся в ней электроны, в определённом направлении. Заряд, накопленный под одним электродом, в любой момент может быть перенесен под соседний электрод, если его потенциал будет увеличен, в то время как потенциал первого электрода, будет уменьшен. Так происходит перемещение по одной строке матрицы.

Если речь идёт о ПЗС-линейке, то заряд в её единственной строке «перетекает» к выходным каскадам усиления и там преобразуется в уровень напряжения на выходе микросхемы.

У матрицы же, состоящей из многих видеострок, заряд из выходных элементов каждой строки оказывается в ячейке ещё одного сдвигового устройства, устроенного обычно точно таким же образом, но работающего на более высокой частоте сдвига. Для использования ПЗС в качестве светочувствительного устройства часть электродов изготавливается прозрачной.

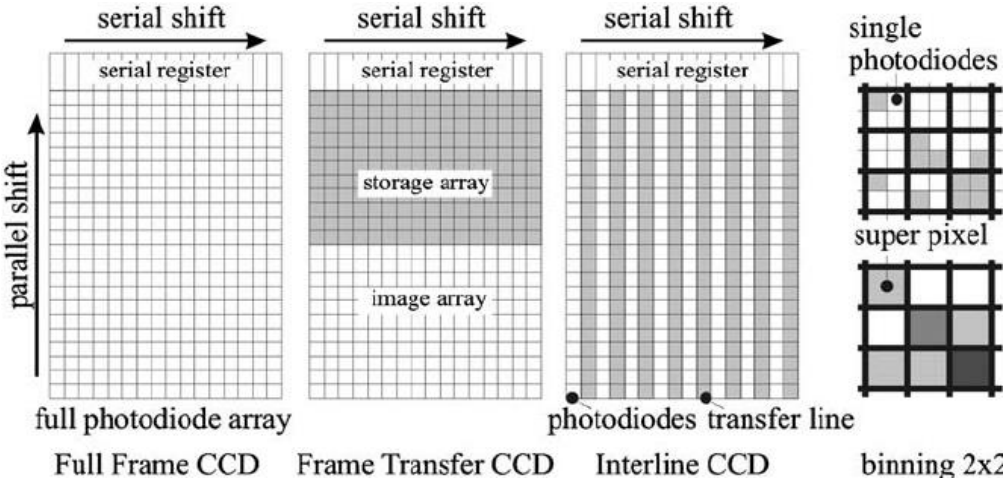

Рис. 1 Виды матриц по принципу перемещения и считывания заряда

По принципу перемещения и считывания заряда различают светочувствительные ПЗС-матрицы с межрядным переносом (Interline Transfer), с покадровым переносом (Frame) и с полнокадровым переносом (Full Frame)(см. Рис. 1).

1) Матрицы с полнокадровым переносом (Full Frame)

Данный тип сенсора является наиболее простым с конструктивной точки зрения и именуется ПЗС-матрицей с полнокадровым переносом (full frame CCD matrix). Помимо микросхем «обвязки», такой тип матриц нуждается также в механическом затворе, перекрывающем световой поток после окончания экспонирования. До полного закрытия затвора считывание зарядов начинать нельзя — при рабочем цикле параллельного регистра сдвига к фототоку каждого из его пикселов добавляются лишние электроны, вызванные попаданием фотонов на открытую поверхность ПЗС-матрицы. Данное явление называется «размазыванием» заряда в полнокадровой матрице (full frame matrix smear).

2) Матрицы с покадровым переносом (Frame Transfer)

Существует усовершенствованный вариант полнокадровой матрицы, в котором заряды параллельного регистра не поступают построчно на вход последовательного, а «складируются» в буферном параллельном регистре. Данный регистр расположен под основным параллельным регистром сдвига, фототоки построчно перемещаются в буферный регистр и уже из

него поступают на вход последовательного регистра сдвига. Поверхность буферного регистра покрыта непрозрачной (чаще металлической) панелью, а вся система получила название матрицы с буферизацией кадра (frame—transfer CCD)

3) Матрицы с межрядным переносом (Interline Transfer)

Специально для видеотехники был разработан новый тип матриц, в котором интервал между экспонированием был минимизирован не для пары кадров, а для непрерывного потока. Разумеется, для обеспечения этой непрерывности пришлось предусмотреть отказ от механического затвора.

Фактически данная схема, получившая наименование матрицы с буферизацией столбцов (interline CCD matrix), в чём-то сходна с системами с буферизацией кадра — в ней также используется буферный параллельный регистр сдвига, ПЗС-элементы которого скрыты под непрозрачным покрытием. Однако буфер этот не располагается единым блоком под основным параллельным регистром — его столбцы «перетасованы» между столбцами основного регистра. В результате рядом с каждым столбцом основного регистра находится столбец буфера, а сразу же после экспонирования фототоки перемещаются не «сверху вниз», а «слева направо» (или «справа налево») и всего за один рабочий цикл попадают в буферный регистр, целиком и полностью освобождая потенциальные ямы для следующего экспонирования.

Попавшие в буферный регистр заряды в обычном порядке считываются через последовательный регистр сдвига, то есть «сверху вниз». Поскольку сброс фототоков в буферный регистр происходит всего за один цикл, даже при отсутствии механического затвора не наблюдается ничего похожего на «размазывание» заряда в полнокадровой матрице. А вот время экспонирования для каждого кадра в большинстве случаев по продолжительности соответствует интервалу, затрачиваемому на полное считывание буферного параллельного регистра. Благодаря всему этому появляется возможность создать видеосигнал с высокой частотой кадров — не менее 30 кадров секунду.

1.3 КМОП-матрицы

КМОП-матрица — светочувствительная матрица, выполненная на основе КМОП-технологии. КМОП- комплементарная структура металл-оксид-полупроводник, набор полупроводниковых технологий построения интегральных микросхем и соответствующая ей схемотехника микросхем. КМОП-матрицах используются полевые транзисторы с изолированным затвором с каналами разной проводимости.

В конце 1960-х гг. многие исследователи отмечали, что структуры КМОП (CMOS) обладают чувствительностью к свету. Однако приборы с зарядовой связью обеспечивали настолько более высокую светочувствительность и качество изображения, что матрицы на технологии КМОП не получили сколько-нибудь заметного развития.

В начале 1990-х характеристики КМОП-матриц, а также технология производства были значительно улучшены. Прогресс в субмикронной фотолитографии позволил применять в КМОП-сенсорах более тонкие соединения. Это привело к увеличению светочувствительности за счёт большего процента облучаемой площади матрицы.

Переворот в технологии КМОП-сенсоров произошёл, когда в лаборатории реактивного движения (Jet Propulsion Laboratory — JPL) NASA успешно реализовали Active Pixel Sensors (APS) — активно-пиксельные датчики (см. Рис.). Теоретические исследования были выполнены ещё несколько десятков лет тому назад, но практическое использование активного сенсора отодвинулось до 1993 года. APS добавляет к каждому пикселю транзисторный усилитель для считывания, что даёт возможность преобразовывать заряд в напряжение прямо в пикселе. Это обеспечило также произвольный доступ к фотодетекторам наподобие реализованного в микросхемах ОЗУ.

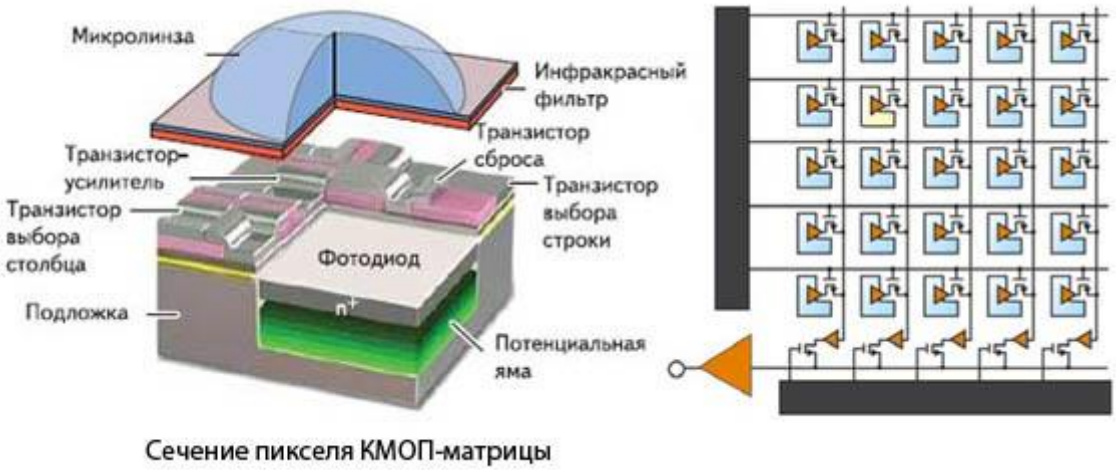

Рис. 2

Пиксель КМОП-матрицы на основе технологии ASP

В отличие от ПЗС-матриц, КМОП-матрица содержат отдельный транзистор в каждом светочувствительном элементе (пикселе) в результате чего преобразование заряда выполняется непосредственно в пикселе.

Синхронизация работы матрицы осуществляется через адресные шины столбцов и строк.

Благодаря такой схеме появляется возможность считывать заряд сразу из группы пикселей (а не последовательно ячейка за ячейкой, как в ПЗС-матрице) или даже выборочно из отдельных пикселей. В такой матрице отсутствует

необходимость в регистрах сдвига столбцов и строк, что намного убыстряет процесс считывания информации с матрицы. Значительно уменьшается и энергопотребление матрицы.

КМОП-матрицы, которые производятся сейчас, делаются на основе датчиков активных пикселей ASP. Датчик активного пикселя (APS) — это датчик изображения, в котором каждая элементарная ячейка датчика пикселей имеет фотоприемник (обычно фотодиод) и один или несколько активных транзисторов. В датчике с активными пикселями металл-оксид-полупроводник (МОП) в качестве усилителей используются полевые МОП-транзисторы (см. Рис.). Существуют различные типы APS, в том числе ранняя APS NMOS и гораздо более распространенная дополнительная MOS (CMOS) APS, также известная как датчик CMOS , которая широко используется в технологии цифровых камер, такие как камеры для мобильных телефонов , веб-камеры , большинство современных цифровых карманных камер, большинство цифровых зеркальных фотокамер с одним объективом (DSLR) и беззеркальные камеры со сменными объективами (MILC). КМОП-датчики появились в качестве

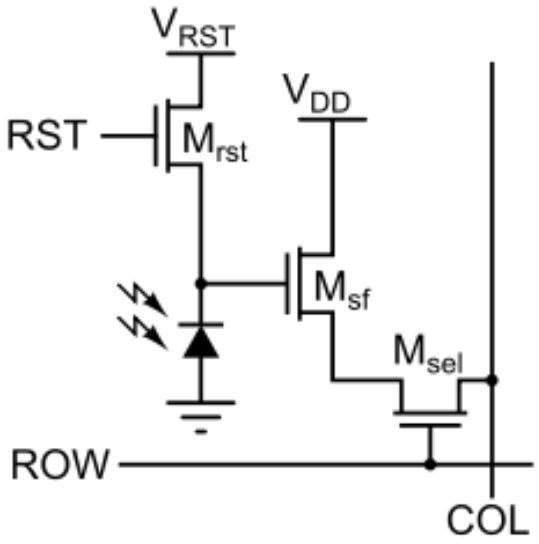

Рис. 3
Трехтранзисторный пиксель APS

альтернативы датчикам изображений с зарядовой связью (ПЗС).

Стандартный КМОП APS пиксели сегодня состоит из фотоприемника, плавающего затвора, а также так называемые 4T клеток, состоящие из четырех КМОП транзистора, включая транзистор передачи, элемент сброса, элемент выбора и транзистор считывания источника-повторителя. Закрепленный фотодиод первоначально использовался в ПЗС с межрядным переносом из-за его низкого темнового тока и хорошего синего отклика, а при соединении с передающим затвором обеспечивает полную передачу заряда от закрепленного фотодиода к плавающему затвору, устраняя запаздывание. Использование

внутрипиксельной передачи заряда может обеспечить более низкий уровень шума, позволяя использовать коррелированную двойную выборку (CDS).

Существует два типа структур с активным пиксельным сенсором (APS): боковой APS и вертикальный APS. Боковая конструкция APS описывается как структура, которая имеет часть области пикселей, используемую для фотодетектирования и хранения сигнала, а другая часть используется для активного транзистора.

Преимущество этого подхода по сравнению с вертикально интегрированной APS состоит в том, что процесс изготовления более прост и в высокой степени совместим с современными процессами CMOS и CCD-устройств.

Вертикальная структура APS — это структура, которая увеличивает коэффициент заполнения (или уменьшает размер пикселя), сохраняя заряд сигнала на выходном транзисторе.

1.4 Типы датчиков на основе светочувствительных матриц

В настоящее время для научных задач исследования космоса и в задачах дистанционного зондирования Земли все большее значение принимает концепция использования малых спутников и аппаратов. Массогабаритные требования, предъявляемые к служебным системам, в том числе приборам астронавигации, становятся ключевыми при разработке таких устройств. Особенно это характерно для аппаратуры наноспутников, малых аппаратов с собственной массой порядка десятков килограммов. Применение в таких спутниках традиционных высокоточных систем ориентации массой в килограмм и более становится проблематичным. Значительное, в несколько раз, уменьшение энергопотребления и массогабаритных параметров отдельного датчика ориентации позволяет объединить такие датчики в систему, что дает возможность по-иному взглянуть на систему ориентации космического аппарата. Установка нескольких малогабаритных датчиков на космический аппарат снимает много проблем с режимом управления и увеличивает надежность всего космического аппарата. Миниатюризация приборов астроориентации обеспечивается, прежде всего, современными технологиями и некоторым снижением требований к точности и чувствительности отдельного датчика.

Исходя из этого факта рассмотрим компактные типы датчиков с небольшой энергозатратностью. Наиболее подходящими под описание конструкции являются датчики направления на Солнце, состоящие из линейчатых ПЗС- матриц и малогабаритных КМОП-матриц.

Первыми рассмотрим конструкции из линейчатых ПЗС-матриц. Они могут быть представлены как 2 перпендикулярно расположенные ПЗС-линейки с 1 отверстием в каждой крышке или 1 ПЗС-линейки с 3 отверстиями в крышке.

1) Конструкция с одной ПЗС-линейкой и тремя отверстиями в крышке.

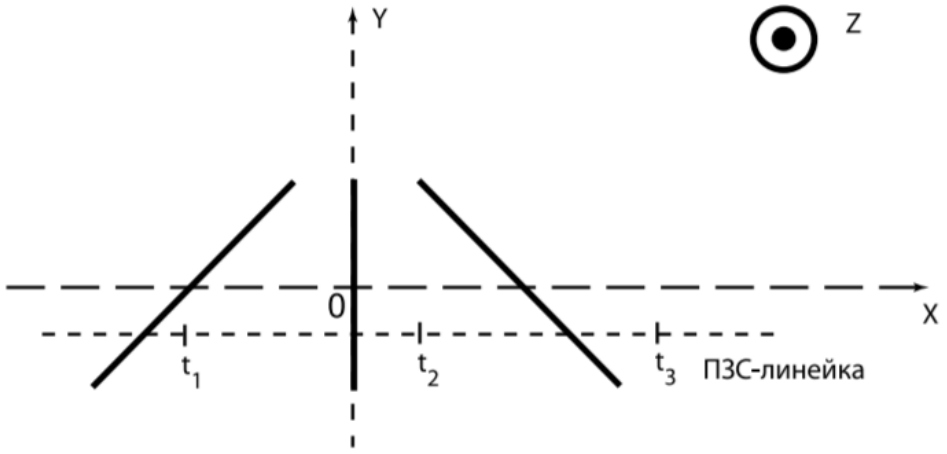

Рис. 4 Расположение щелей оптического солнечного датчика: t1, t2, t 3 — координаты энергетических центров изображений трех щелей кодирующей маски

В оптическом элементе имеются три щели (см. Рис.) при этом крайние щели образуют с центральной щелью угол 45°. Поверхность оптического элемента и центральная щель задают внутреннюю систему координат солнечного датчика. Солнечное излучение, проходя через оптический элемент, формирует изображение трех щелей на чувствительной поверхности ПЗС-линейки. По положению центральной группы щелей на ПЗС-линейке относительно центрального пиксела линейки можно определить угол Солнца в плоскости матрицы (см. Рис.), по расположению изображений крайних щелей относительно изображения центральной щели можно определить угол Солнца в плоскости матрицы. В принципе, для определения направления на Солнце достаточно изображений щелей одной крайней и центральной группы, но для увеличения угла поля зрения

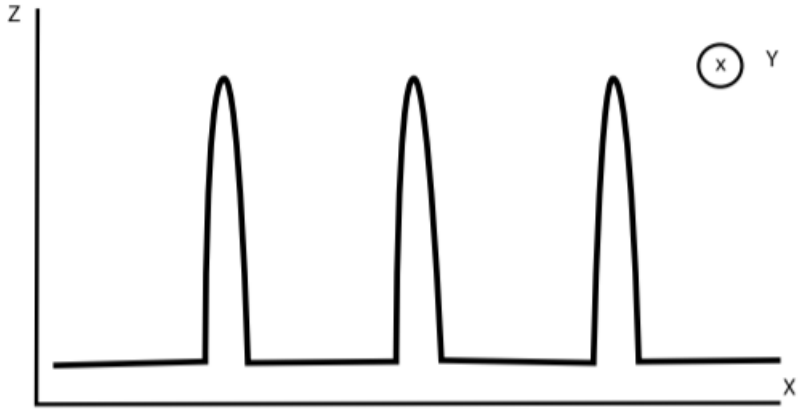

Рис. 5 Изображение щелей на ПЗС-линейке

используют три щели.

2) Две ПЗС линейки расположенные перпендикулярно друг другу.

Каждая ПЗС линейка вычисляет угол падения лучей света относительно одной оси. Происходит это путем прохождения света через щель в крышечке, которой накрыты матрицы, и попаданием его на активную часть светочувствительной матрицы. После чего они детектируются и просчитывается их местоположение на матрице. Затем высчитывается угол падения солнечных лучей и с помощью двух таких углов в двух перпендикулярных плоскостях мы высчитываем угол наклона Солнца относительно поверхности светочувствительной матрицы.

Угол обзора в таком случае составляет порядка ±60°.

Теперь рассмотрим КМОП-матрицы. Существует два основных типа конструкций — это две линейные КМОП-матрица с одним отверстием в каждой крышечке и прямоугольная КМОП-матрица так же с одним отверстием в крышечке.

1) В случае, когда мы используем две линейки КМОП-матриц они располагаются перпендикулярно друг другу (см. Рис.). Над каждой матрицей есть крышечка с продольным отверстием, через которое попадает солнечный свет. Тем самым каждая при вычислении каждая линейка даст результат по одной оси. После этого остается высчитать угол наклона относительно Солнца основываясь на два угла в двух плоскостях. Средний угол обзора по одной оси которого можно добиться это ±45°, однако это

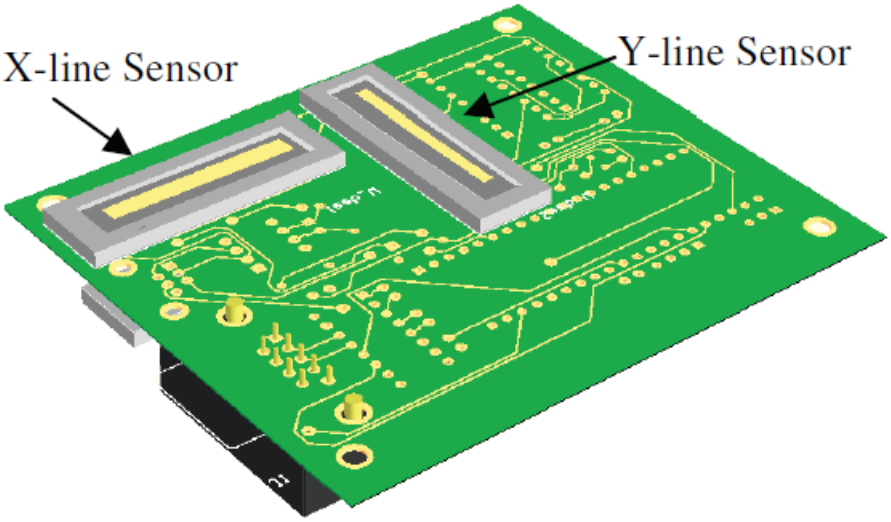

Рис. 6 Расположение двух КМОП-линеек на плате

еще зависит от размеров матрицы и пикселей.

2) Когда у нас одна КМОП-матрица она должна воспринимать световые лучи сразу по двум осям. Отсюда можно сделать вывод, что чем ближе форма матрицы к квадрату, тем более равные углы обзора по двум координатам она будет иметь. Матрица накрывается крышечкой с небольшим отверстием, через которую на нее будет попадать солнечный свет (см. Рис.). Солнечные лучи проходят через отверстие и обнаруживаются двумерным массивом КМОП. После чего он детектирует и запоминает куда упал свет. Пиксельные позиции вычисляются, а затем устанавливается их положение по двум осям. И исходя из этих данных рассчитывается угол падения солнечного луча. В конструкциях такого типа можно легко достичь угла обзора в ±75°.

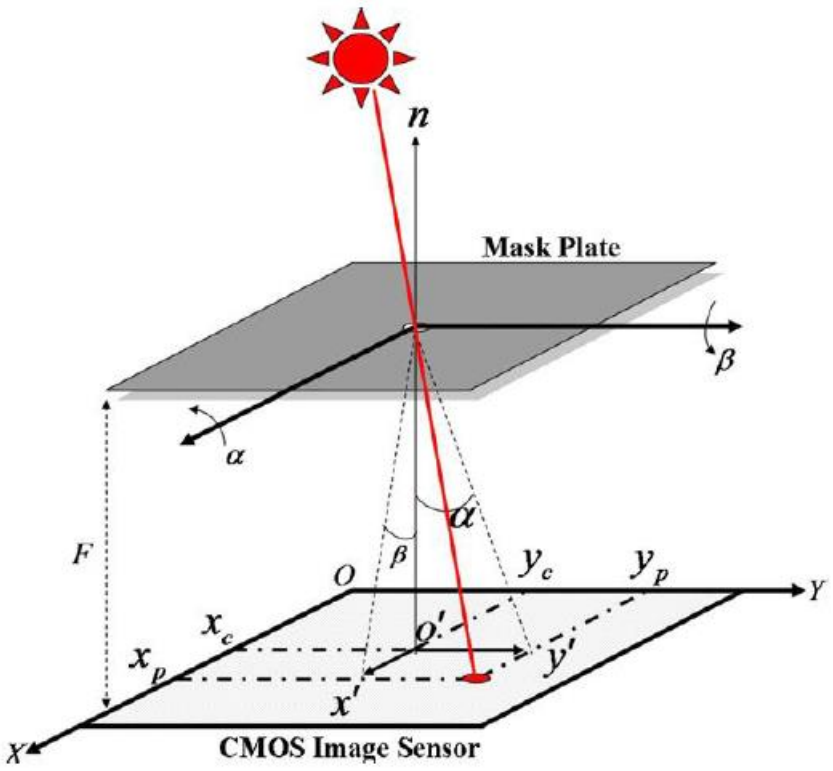

Рис. 7 Принцип работы КМОП-матрицы для датчика направления на солнце

Рассмотрев эти конструкции, мы можем сделать вывод, что структура, состоящая из одной КМОП-матрицы с одним отверстием в крышечке, выдает наибольший угол обзора.

ГЛАВА 2
ГЕНЕРАЦИЯ ИЗОБРАЖЕНИЯ НА МАТРИЦЕ

2.1 Информация о используемой матрице

В данной работе использовалась модель КМОП-матрицы MT9V032, позволяющая получить изображение падающего на него луча.
Общий вид КМОП датчика (см. Рис. 8).

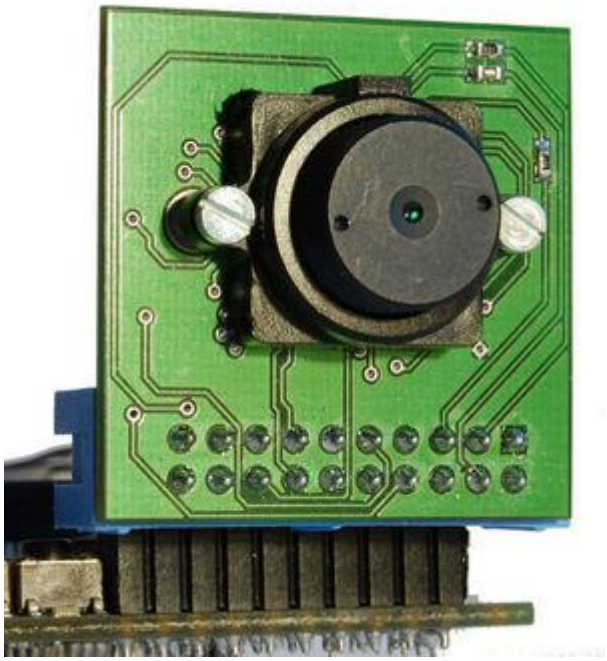

Рис. 8 Вид КМОП матрицы MT9V032

Характеристики используемого датчика:

- Оптический формат: 1/3 дюйма;
- Активный размер тепловизора: 4.51мм (В) × 2.88мм (Ш);
- Диагональ: 5.5мм;
- Активные пиксели: 752H × 480V;
- Массив цветных пикселей монохромный или цветной RGB;
- Тип затвора: глобальный затвор???????;
- Максимальная скорость передачи данных: 26.6 MPS / 26.6 MHz;
- Разрешение: 752 × 480;
- Частота кадров в секунду (при полном разрешении): 60;
- Чувствительность 4.8 V/lux-sec (550nm);
- Динамический диапазон > 55дБ; > 80 дБ – 100 дБ в режиме HDR;
- Напряжение питания 3,3 В ± 0,3 В;
- Рабочая температура от -30 ° С до + 70 ° С;

- Потребляемая мощность <320 мВт при максимальной скорости передачи данных;
- 100 Вт в режиме ожидания.

Особенности матрицы:

- Формат массива: Wide-VGA, активный 752H x 480V (360,960 пикселей)
- Пиксели фотодиодов с глобальным затвором; Одновременная интеграция и считывание
- Монохромный или цветной: улучшенные характеристики в ближнем ИК-диапазоне для использования с невидимой подсветкой в ближнем ИК-диапазоне
- Режимы считывания: прогрессивный или чересстрочный
- Эффективность затвора:> 99%
- Простой двухпроводной последовательный интерфейс
- Возможность блокировки регистра
- Размер окна: программируется пользователем на любой меньший формат (QVGA, CIF, QCIF и т. Д.). Скорость передачи данных может поддерживаться независимо от размер окна
- АЦП: на кристалле, 10-битный параллельный столбец (опция для работы в режиме компандирования с 12-битного до 10-битного);
- Автоматическое управление: автоматическая регулировка экспозиции (AEC) и автоматическое усиление; Контроль (AGC); Переменный региональный и переменный вес AEC / AGC

2.2 Теория генерации изображения

Как видно на рисунке падения луча на матрицу (см. Рис.), устройство состоит из двух частей: активного датчика КМОП и маски. Маска представляет собой круглое отверстие, через которое может фильтровать солнце: активный

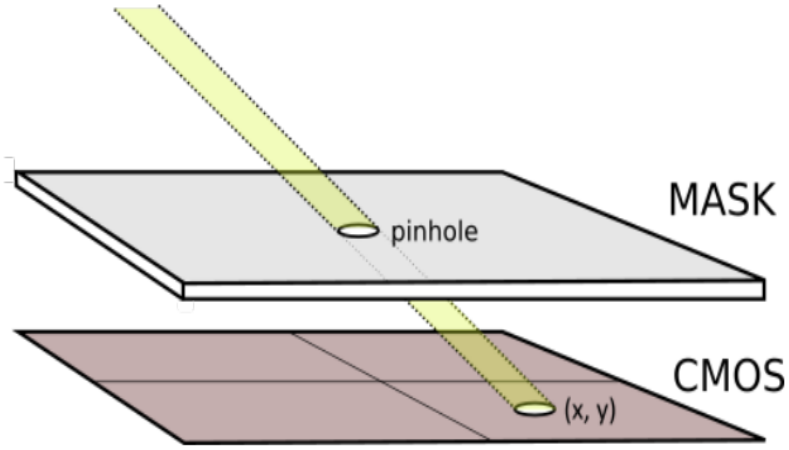

Рис. 9 Падение луча на матрицу

пиксель (x, y) (см. Рис.). Датчик используется как детектор светового пятна.

Зная координаты точки (x, y) на КМОП можно использовать простую тригонометрию для определения азимутального и зенитного углов. (см. Рис.).

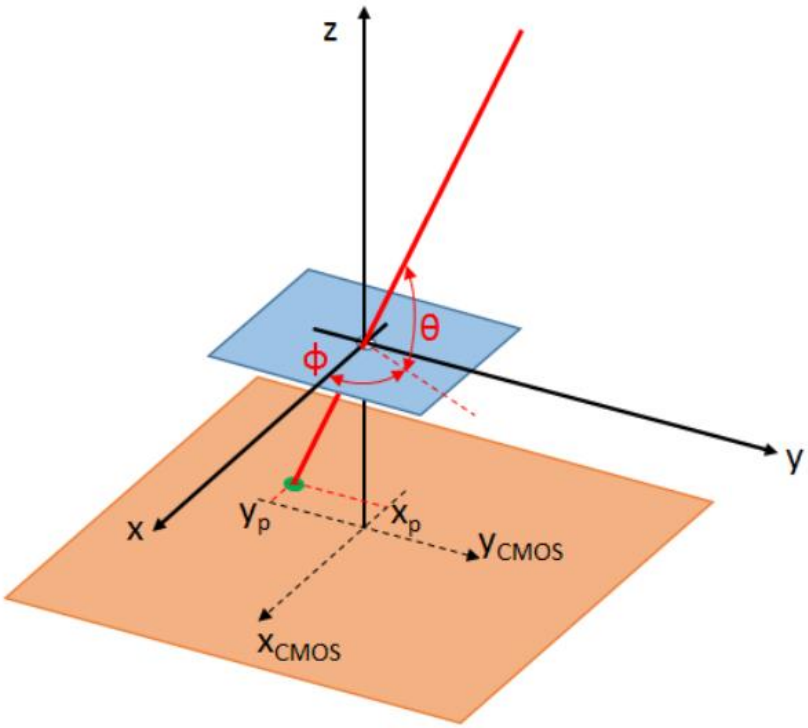

Рис. 10 Базовая геометрия датчика: координаты пятна определяются азимутальным и зенитным углами.

2.3 Геометрическая модель и моделирование

Геометрическая модель предлагаемого датчика представлена на рисунке 10 (см. Рис.), с дополнением, что синяя плоскость – маска, оранжевая – КМОП матрица. Как упоминалось ранее, датчик измеряет направление вектора солченых лучей, т.е. относительное положение солнца в поле зрения. Эта информация может быть представлена как и азимутальным углом, так и зенитным или же связанным вектором

$$v = (\cos\Phi\cos\theta, \sin\Phi\cos\theta, \sin\theta), \quad (1)$$

где Ф– азимутальный угол, θ– зенитный угол

Но вся эта информация может быть получена, зная положение светового пятна (x, y), поскольку расстояние h между маской и КПОМ мы знем:

$$v = \frac{(x,y,h)}{\sqrt{(x^2+y^2+x^2}}, \quad (2)$$

где x, y – координаты пятна на датчике, h – высота маски над датчиком

Эта формулировка не включает никаких тригонометрических функций, а единственное и реальное решение. существует для любого положения светового пятна на КМОП.

Зная размер КМОП и расстояние установки маски, можно определить теоретическое поле зрения датчика, учитывая идеальную маску с незначительной толщиной и отсутствием дифракция. Для датчика, описанного в

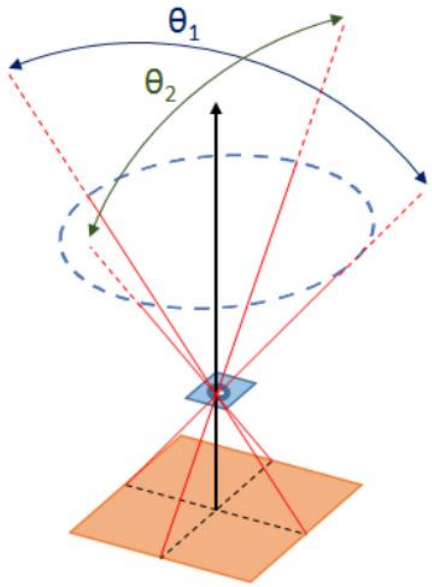

Рис. 11 Поле зрения датчика

этой работе, два угла поля зрения θ1 и θ2 представленные на рисунке 11 составляют соответственно 64,1 ° и 49,0 °. (см. Рис.).

2.4 Моделирование падающего луча

Вышеупомянутое геометрическое описание позволило разработать три различные модели возрастающей сложности, анализируя реакцию датчика солнца на поступающее излучение, как показано на рисунке 5. В идеальном случае маски без толщины проецируемое световое пятно имеет точный размер и форму отверстия маски; Измеряя центр светового пятна, можно напрямую рассчитать вектор направления Солнца.

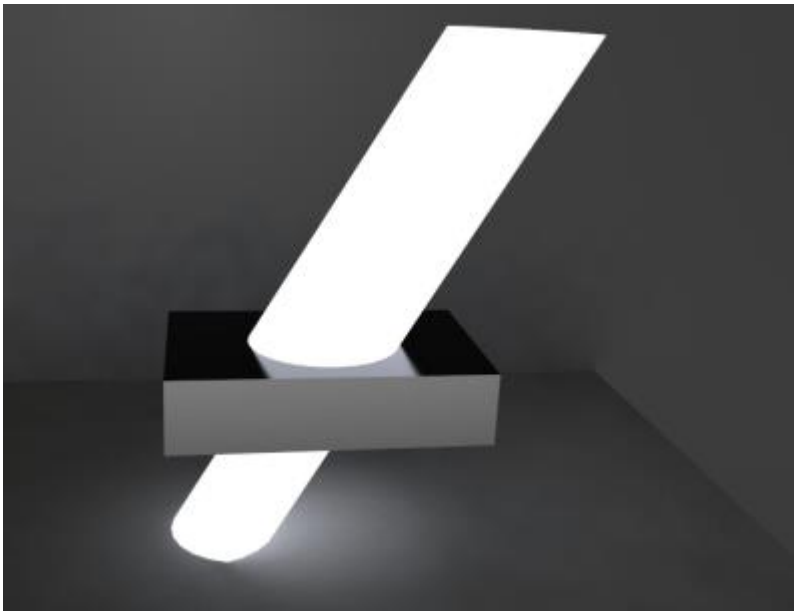

Рис. 12 Влияние толщины маски на проецируемое пятно света.

Но, пока толщиной маски можно пренебречь, такое поведение солнечного луча возможно. Маска в данном устройстве, имеет толщину, сопоставимую с диаметром отверстия, поэтому часть входящего излучения задерживается границами маски, изменяя форму кругового пятна, что можно заметить на рисунках 13(см. Рис.) при θ1 = 90, Ф1 = 90, без обрезания краёв пятна, и при θ2 = 78, Ф2 = 78. На рисунках видно, что при изменении задаваемых углов, форма круга изменятся к более эллипсоидной и смещается от центра.

Так же следует отметить, что обе модели выше были разработаны с использованием идеального источника света, то есть точечный источник с параллельными падающими лучами.

Программная реализация отображения пятна на матрице предназначена для оценки перевода и расчёта правильной ориентации пятна. Это моделирование можно провести, отметив, что геометрия задачи допускает

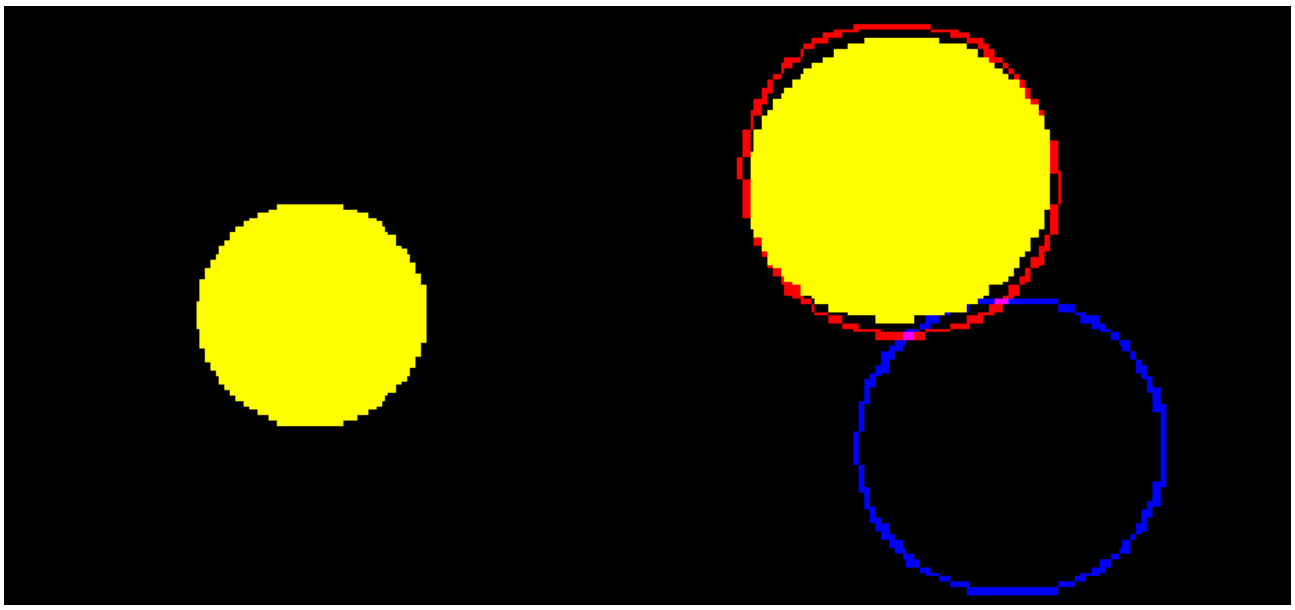

Рис. 13 Световые пятна при θ1 = 90, Ф1 = 90 (слева) и θ2 = 78, Ф2 = 78 (справа). Красные границы обозначают исходную форму пятна, а синий - позицию

осесимметричные упрощения. Кусочное уравнение, описывающее форму проецируемого пятна в полярных координатах (далее выраженных в терминах азимута и зенита), будет иметь следующий вид:

$$f(\phi,\theta)\begin{cases} r(\theta) - 2r(\theta)(x_0\cos\phi + y_0\sin\phi) + {x_0}^2 + {y_0}^2 = d,\ R_{c0} - \frac{d}{2} < r < R_{c0} + \frac{\Delta C}{2} \\ r(\theta) - 2r(\theta)(x_1\cos\phi + y_1\sin\phi) + {x_1}^2 + {y_1}^2 = d,\ R_{c1} - \frac{\Delta C}{2} < r < R_{c1} + \frac{d}{2} \end{cases}, \quad (3)$$

$$r(\theta) = (t+h) * \tan(\frac{\pi}{2} - \theta) \quad (4)$$

$$\Delta C = h * \tan(\frac{\pi}{2} - \theta) \quad (5)$$

$$\begin{cases} R_{c0} = \sqrt{{x_0}^2 + {y_0}^2} \\ R_{c1} = \sqrt{{x_1}^2 + {y_1}^2} \end{cases} \quad (6)$$

$$\begin{cases} x_1 = \Delta C * \cos(\phi + \pi) \\ y_1 = \Delta C * sin(\phi + \pi) \end{cases} \quad (7)$$

$$\begin{cases} x_0 = \Delta(t+h) * \tan\left(\frac{\pi}{2} - \theta\right) * \cos(\phi + \pi) \\ y_0 = (t+h) * \tan(\frac{\pi}{2} - \theta) * sin(\phi + \pi) \end{cases} \quad (8)$$

где d – диаметр отверстия в маске, t – толщина отверстия, h –высота отверстия над матрицей, x_1, y_1, x_0, y_0 – координаты центров двух окружностей, формирующих финальное изображение пятна, ΔC – расстояние между центрами окружностей.

На рисунке 14 (см. Рис.) можно увидеть схематическое представление изменённого солнечного пятна на матрице.

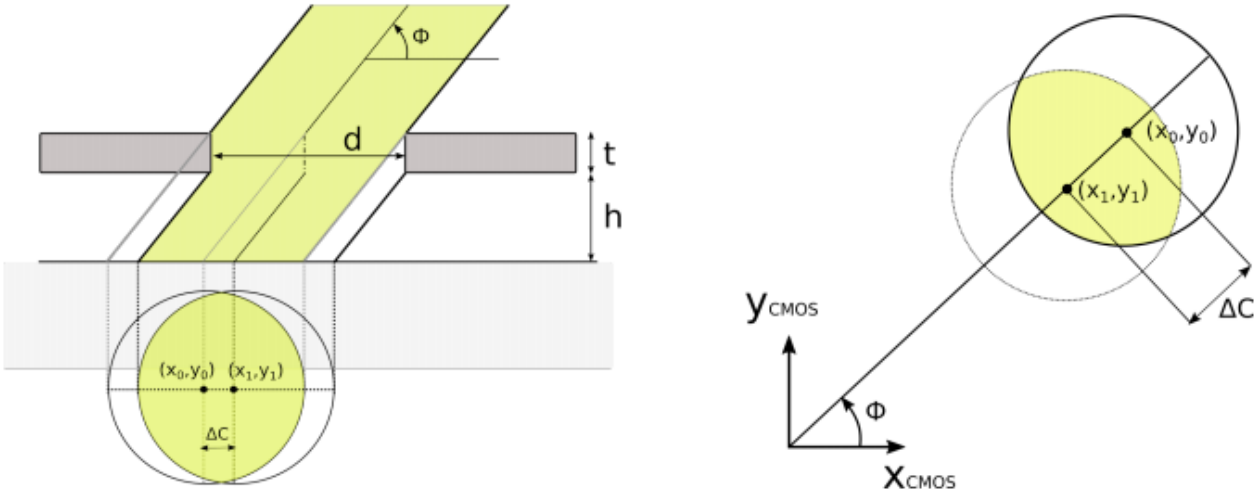

Рис. 14 Схематическое представление влияние толщины отверстия на финальное изображение

ГЛАВА 3
ОБРАБОТКА ИЗОБРАЖЕНИЙ ПЯТНА

В данной главе будет рассмотрен процесс поиска пятна на матрице несколькими методами и их центра, сравнение их эффективности.

Процесс нахождения углов падения луча состоит из нескольких частей:

1. Поиск пятна на матрице
2. Локализация пятна на матрице
3. Поиск центра пятна

3.1 Поиск пятна

Первым делом проводится бинаризация картинки пятна.
Процесс поиска пятна реализовывался двумя способами:

1. Brute Force — долгий перебор всех пикселей матрицы и сравнение цвета каждого пикселя, их сохранение в массив для последующих действий над этими точками;
2. Случайным образом выбирается точка на матрице до тех пор, пока не попадётся хотя бы одна закрашенная; Точки в которые случайно попали, но там не было пятна – запоминаем, либо исключаем возможность случайно получить его ещё раз, т.к. если матрица будет пустая, случайные точки будут выбираться бесконечно.
3. Brute Force+ & Rand+ - в данных алгоритмах происходит тоже самое, что и в алгоритмах описанных выше кроме того, что после того, как всё пятно будет обработано и записано алгоритм прекратит работать, т.е. не будет проверять чёрную матрицу до самого конца или квадрат для случая с rand, что значительно ускорит работу алгоритма Brute Force

3.2 Локализация пятна

Процесс локализации пятна необходим для того чтобы получить все пиксели солнечного пятна на изображении, для последующей обработки.

1. Т.е. метод Brute Force подразумевает перебор всех пикселей матрица на данном этапе объединяется поиск пятна и локализация пятна;

2. При попадании случайной точки в пятно, для локализации пятна достаточно изучить квадрат размерами [d x d], где d – диаметр отверстия маски. Это обусловлено тем, что при прямом падении луча, $\theta = 0$, $\Phi = 0$, изображение пятна не искажается т.к. не перекрывается стенками

отверстия и мы получаем пятно максимально возможного размера. Т.к. пятна по размерам больше чем этого быть не может достаточно изучить 4 квадрата размерами [d/2 x d/2], что можно увидеть на рисунке 15(см. Рис.

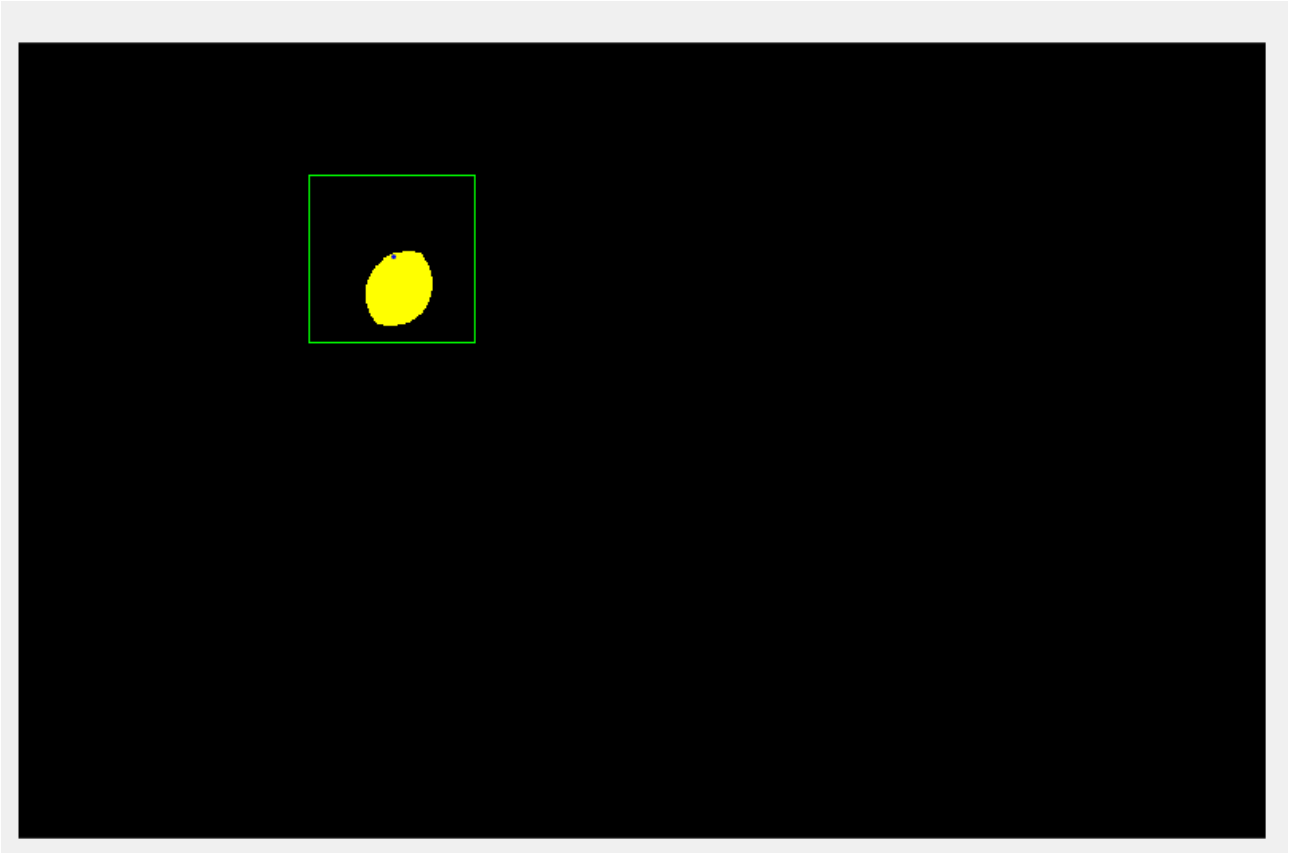

Рис. 15 Локализация пятна после случайного обнаружения пикселя пятна

) ниже. На изображении обозначена синяя маленькая точка, в которую попал алгоритм при случайном поиске и от этой позиции берутся 4 другие позиции, такие, чтобы в квадрат полученный между исходной точкой и любой из 4, влез круг максимального размера, так мы будем уверен, что пятно точно в этой области.

Далее в данном алгоритме так же происходит перебор каждого пикселя внутри выделенного квадрата, но по сравнению с обычным перебором в среднем мы перебираем гораздо меньше позиций матрицы, при этом добавляя их в некоторый массив.

3.3 Поиск центра пятна

Для обоих методов поиск центра пятна производится одинаково – координаты, полученные после локализации пятна, усредняются по формулам (9), (10)

$$x_c = \frac{\sum_{ellipce} x_i}{n_x}, \quad (9)$$

$$y_c = \frac{\sum_{ellipce} y_i}{n_y}, \quad (10)$$

где x_c, y_c – координаты центра пятна на датчике, n_x, n_y - количество найденных точек пятна, x_i, y_i – точка пятна.

3.4 Вычислительная эффективность

Для проверки вычислительной эффективности двух алгоритмов я програмно генерировал 1000 изображений пятна со случайными параметрами азимутальных и зенитных углов, а после каждое из изображений обрабатывал програмно двумя алгоритмами.

Для оценки эффективности воспользовался встроенной в matlab функцией “Run and Time” для анализа обработки 1000 изображений. Результаты обработки 1000 изображений можно увидеть на рисунках 16 -18 ((см. Рис.), (см. Рис.), (см.

Profile Summary (Total time: 1198.495 s)

Generated 28-дек.-2020 04:17:47 using performance time.

Function Name	Calls	Total Time (s)	Self Time* (s)	Total Time Plot (dark band = self time)
brutForce	1000	948.949	618.378	
squeeze	384715208	352.585	352.585	
writeXLSFile	1000	169.045	136.374	
randPick	1000	72.540	50.514	
createWorkbook>constructWorkbook	1000	32.094	32.094	
imagesci\private\pngreadc (MEX-file)	1000	2.579	2.579	
ImageProcessing	1	1198.466	0.841	
imagesci\private\readpngutil	1000	3.184	0.547	
writetable	1000	169.970	0.356	
parseArgs	3000	0.313	0.313	
cell.strcat	3000	0.268	0.251	
imread>get_full_filename	1000	0.236	0.236	
writecell	1000	171.271	0.235	
int2str	5000	0.231	0.231	
rgb2gray	1000	0.249	0.218	
table.table>table.table	1000	0.931	0.188	
imread	1000	3.883	0.161	
makeUniqueStrings>makeUnique	2000	0.137	0.137	
string.strcat	3000	0.396	0.127	
num2str	5000	0.326	0.095	
writeXLSFile>getSheetFromBook	1000	0.093	0.093	
imread>call_format_specific_reader	1000	3.300	0.092	
makeUniqueStrings	2000	0.294	0.089	

Рис. 16 суммарный отчёт обработки 1000 изображений

Рис.)), красным отмечены использованные в алгоритмах функции (функции загрузки изображения/записи в файл и т.д. не учитывались)

brutForce (Calls: 1000, Time: 948.949 s)

Generated 28-дек.-2020 04:19:20 using performance time.
Function in file C:\Users\yarba\Desktop\ДО\course_4\code\brutForce.m
Copy to new window for comparing multiple runs

Parents (calling functions)

Function Name	Function Type	Calls
ImageProcessing	Script	1000

Lines that take the most time

Line Number	Code	Calls	Total Time (s)	% Time	Time Plot
33	`if(squeeze(imageGS(j, i, :)) ~= 0)`	360960000	921.942	97.2%	
37	`end`	360960000	13.645	1.4%	
38	`end`	360960000	12.373	1.3%	
34	`coloredX(end + 1) = i;`	1484025	0.386	0.0%	
35	`coloredY(end + 1) = j;`	1484025	0.269	0.0%	
All other lines			0.334	0.0%	
Totals			948.949	100%	

Children (called functions)

Function Name	Function Type	Calls	Total Time (s)	% Time	Time Plot
squeeze	Function	360960000	330.565	34.8%	
deg2rad	Function	1000	0.006	0.0%	
Self time (built-ins, overhead, etc.)			618.378	65.2%	
Totals			948.949	100%	

Рис. 17.а отчёт обработки 1000 изображений, метод Brute Force

< 0.001	1000	31	`for  i = 1 : columns`
0.077	752000	32	`    for j = 1 : rows`
921.942	360960000	33	`        if(squeeze(imageGS(j, i, :)) ~= 0)`
0.386	1484025	34	`            coloredX(end + 1) = i;`
0.269	1484025	35	`            coloredY(end + 1) = j;`
0.046	1484025	36	`            isFlag = true;`
13.645	360960000	37	`        end`
12.373	360960000	38	`    end`
0.076	752000	39	`end`

Рис. 17.б Brute Force функция сработавшая больше всего раз

randPick (Calls: 1000, Time: 72.540 s)

Generated 28-дек.-2020 04:20:57 using performance time.
Function in file C:\Users\yarba\Desktop\ДО\course_4\code\randPick.m
Copy to new window for comparing multiple runs

Parents (calling functions)

Function Name	Function Type	Calls
ImageProcessing	Script	1000

Lines that take the most time

Line Number	Code	Calls	Total Time (s)	% Time	Time Plot
120	`if(squeeze(imageGS(j, i, :)) ~= 0)`	22457164	58.316	80.4%	
57	`if(squeeze(imageGS(randY, randX, :)) ~= 0)`	1298044	3.988	5.5%	
45	`if (any(repeat(:) == 0))`	15529	2.097	2.9%	
41	`randX = randi(columns, 1);`	1313573	1.860	2.6%	
42	`randY = randi(rows, 1);`	1313573	1.451	2.0%	
All other lines			4.829	6.7%	
Totals			72.540	100%	

Children (called functions)

Function Name	Function Type	Calls	Total Time (s)	% Time	Time Plot
squeeze	Function	23755208	22.020	30.4%	
deg2rad	Function	1000	0.007	0.0%	
Self time (built-ins, overhead, etc.)			50.514	69.6%	

Рис. 18.а отчёт обработки 1000 изображений, метод Rand

По полученной статистике видно, что основное время работы алгоритмов уходит на проверку цвета пикселя squeeze(image);, можно убедится в этом просмотрев результаты работы алгоритмов Brute Force + & Rand+ на рисунке 19 (см. Рис.)

Profile Summary (Total time: 758.974 s)

Generated 28-дек.-2020 05:23:24 using performance time.

Function Name	Calls	Total Time (s)	Self Time* (s)	Total Time Plot (dark band = self time)
brutForce	1000	535.417	348.605	
squeeze	232750171	208.203	208.203	
writeXLSFile	1000	145.393	121.322	
randPick	1000	70.126	48.723	
createWorkbook>constructWorkbook	1000	23.522	23.522	
imagesci\private\pngreadc (MEX-file)	1000	2.649	2.649	
ImageProcessing	1	758.721	0.806	
imagesci\private\readpngutil	1000	3.246	0.540	
writetable	1000	146.263	0.330	

Рис. 19. отчёт обработки 1000 изображений, методами Brute Force+ & Rand+

В данном случае виден огромнейший прирост скорости выполнения алгоритма в около 2 раз, скорость алгоритма Brute Force увеличилась почти в два раза, а количество выполняемых операций проверки уменьшилось на 151978080, что является значительным приростом.

3.5 Точность алгоритмов

Так я провёл проверку точности вычисления алгоритмов, используя результаты всех 1000 картинок, кроме случаев, когда пятно заходило за край матрицы краем либо полностью, т.к. в данном случае метод нахождения центра пятна указанный выше работать не будет.

После проведённых вычислений и расчётов было получено, что:

Среднее отличие координат центра оригинального изображения (полученного генерированием картинки) Xc и Yc от координат центра пятна, полученного методом Brute Force+ Xb и Yb равно:

для координаты **X: 0.698193411264612**, для **Y: 0.608646188850967** , что является достаточно высокой точностью, учитывая что речь идёт о разности координаты матрицы, т.е. в данном случае средняя ошибка вычисления центра в пределах 1 пикселя.

Среднее отличие координат центра оригинального изображения Xc и Yc от координат полученных методом Rand+ Xr и Yr равняется:

Для **X: 0.711252653927813**, **Y: 0.605473204104903.** Т.к. в данном случае ошибка вычисления так же находится в пределах 1 пикселя, то это является достаточно точным результатом для матрицы с таким низким разрешеним.

ЗАКЛЮЧЕНИЕ

В данной работе был рассмотрен один из востребованных в наше время солнечных датчиков. Для модели представленного выше датчика были проведены исследования для создания и моделирования падающего луча на светочувствительную матрицу данного датчика в зависимости от исходных параметров падения луча, а так же характеристик матрицы, т.е. высоты маски, толщины маски, разрешения и т.д.

Был проведёт опыт по созданию 1000 различных изображений с солнечным пятном на матрице в зависимости от случайно заданных азимутальных и зенитных углов, значение центра полученных окружностей, а так же задаваемых углов сохранялись в файл для последующей обработки.

В последней части была проведена обработка всей 1000 до этого сгенерированных картинок для поиска центра пятна несколькими алгоритмами. В данной работе были приведены два алгоритма, один из них использовал случайный выбор точки, а второй перебор, и две вариации этих алгоритмов: обычный и алгоритм+, с окончанием поиска пятна заранее. В результате были получены центры пятен и сохранены в файл для последующей обработки. Так же был проведёт анализ вычислительной эффективности данных алгоритмов поиска в результате которого было совершенно очевидно, что использование rand даёт гораздо более быстрый и менее затратный результат чем, простой перебор всех пикселей.

В результате анализа полученных центров пятен получили, что полученные при помощи алгоритмов значения координат отличаются от оригинальных на менее чем 1, что является хорошим результатом работы алгоритмов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Andrea Antonello, Lorenzo Olivieri, Alessandro Francesconi, Low-Cost, High-Resolution, Self-Powered, Miniaturized Sun Sensor For Space Applications, CISAS G. Colombo, University of Padova, Italy; DII, University of Padova, Italy.

2. Marcello BUONOCORE, Michele GRASSI and Giancarlo RUFINO, Aps-based miniature sun sensor for Earth observation nanosatellites.

3. Jorge Prado-Monila, Hector Arriaga-Arrollo and Julio-Cesar Balanza-Ramagnoli, Low-cost CMOS active pixel Sun Sensor for nanosatellite’s two-axis attitude determination. Instituto de Geografia, Universidad Nacional Autonoma de Mexico, Mexico City, Mexico.

4. Ahad Ali, Fahad Tanveer, Low-Cost Design and Development of 2-Axis Digital Sun Sensor, AOCS Section, Satellite Research & Development Centre-Karachi, (SUPARCO) Pakistan, Journal of Space Technology, 2011. – С.82-87.С.

5. Rufino, G., and Grassi, M., "Digital sun sensor multi-spot operation." Sensors 12.12 (2012): 16451-16465

6. MT9V032 1/3-Inch Wide VGA CMOS Digital Image Sensor, © Semiconductor Components Industries, LLC, 2006 May, 2017 − Rev. 7

https://www.redblobgames.com/pathfinding/a-star/introduction.html

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

КОД MATLAB ДЛЯ МОДЕЛИРОВАНИЯ ПАДАЮЩЕГО ЛУЧА НА МАТРИЦУ (ФАЙЛ SUNBLEND.M)

```
close all;
clear all;
clc;

d = 0.0003;      % диаметр отверстия
t = 0.00005;      % толщина отверстия
h = 0.0007;       % высота отверстия
format long
H = 4.51e-3;     % Размеры матрицы
W = 2.88e-3;
x = zeros(1, 360);
y = zeros(1, 360);
pxH = 752;
pxW = 480;
pxSize = W / pxW;
randMinTh = 1;
randMaxTh = 80;
randMinPhi = -180;
randMaxPhi = 180;

filename = 'compare.xlsx';
fileData = {"Theta", "Phi", "Центр пятна X", "Центр пятна Y"};
writecell(fileData, filename, 'sheet', 1, 'Range', "D1");

for iii = 1 : 10
%     theta_src = 55;      % исходный зенитный угол
%     phi_src = 58;        % исходный азимутальный угол
    theta_src = randMinTh + rand(1, 1) * (randMaxTh - randMinTh);      %
СЛУЧАЙНЫЙ зенитный угол
    phi_src = randMinPhi + rand(1, 1) * (randMaxPhi - randMinPhi);       %
СЛУЧАЙНЫЙ азимутальный угол

    theta = deg2rad(90) - deg2rad(theta_src);      % зенитный угол падения
солнечных лучей
    phi = deg2rad(phi_src);      % азимутальный угол падения солнечных лучей
    dC = h * tan(pi/2 - theta);
    angle = deg2rad(0 : 359);           % вспомогательный массив углов
    x = (d/2) * cos(angle);             % координата x контура пятна
    y = (d/2) * sin(angle);             % координата y контура пятна
    xPx = ceil((x + W/2) / pxSize);  % x - контур пятна в пикселях
    yPx = ceil((y + H/2) / pxSize);  % y - контур пятна в пикселях
    x1 = dC * cos(phi + pi); % x1
    y1 = dC * sin(phi + pi); % y1
    x1Px = ceil((x1 + W/2 + x) / pxSize); % x1 px
    y1Px = ceil((y1 + H/2 + y) / pxSize); % y1 px

    x0 = (t + h) * tan(pi/2 - theta) * cos(phi + pi); % контур смещённого пятна
(без учёта обрезания круга)
    y0 = (t + h) * tan(pi/2 - theta) * sin(phi + pi); % контур смещённого пятна
(без учёта обрезания круга)

    psY = ceil((y1/2 + y0/2 + H/2) / pxSize);
```

```
    psX = ceil((x1/2 + x0/2 + W/2) / pxSize);

    x0Px = ceil((x0 + W/2 + x) / pxSize); % x0Px - контур смещённого пятна (без
учёта обрезания круга) в пикселях
    y0Px = ceil((y0 + H/2 + y) / pxSize); % y0Px - контур смещённого пятна (без
учёта обрезания круга) в пикселях

    % нарисовать круги со смещением и поворотом
    image = zeros(pxW, pxH, 3);

    for i = 1 : length(x0Px)
    %     image(xPx(i), yPx(i), [3 3 3]) = 1;
    %     image(x1Px(i), y1Px(i), [1 2 1]) = 1;
    %     image(x0Px(i), y0Px(i), [1 1 1]) = 1;
    end

    % найти максимумы / минимумы из двух смещённых окружностей
    minX = min(x0Px);
    minY = min(y0Px);
    maxX = max(x1Px);
    maxY = max(x1Px);

    if (minX > min(x1Px))
       minX = min(x1Px);
    end
    if (minY > min(x1Px))
       minY = min(x1Px);
    end
    if (maxX < max(x0Px))
       maxX = max(x0Px);
    end
    if (maxY < max(y0Px))
       maxY = max(y0Px);
    end

    x_ = []; % координата усечённого и перемещённого и повёрнутого круга
    y_ = []; % координата усечённого и перемещённого и повёрнутого круга

    % найти пересечение двух окружностей
    for i = minX : maxX
        for j = minY : maxY

            if (i <= 0 || i > pxW)
                continue;
            end
            if (j <= 0 || j > pxH)
                continue;
            end

            in = inpolygon(i, j, x0Px, y0Px);
            in2 = inpolygon(i, j, x1Px, y1Px);
            if (in ~= 0 && in2 ~= 0)
    %             image2(i, j, [2 2 2]) = 1;
                x_(end + 1) = i;
                y_(end + 1) = j;
            end
        end
    end

    % Закрашивание общей области
    for i = 1 : length(x_)
    %     image(x_(i), y_(i), [1 1 3]) = 1;
```

```
        image(x_(i), y_(i), 1) = 1;     %Red
        image(x_(i), y_(i), 2) = 1;   %Green
        image(x_(i), y_(i), 3) = 0;   %Blue
    end

    imshow(image);
    imwrite(image, strcat("./Images/Acircle_", num2str(iii), ".png"));

    fileData = {theta, phi, xArray, yArray};
    writecell(fileData, filename, 'sheet', 1, 'Range', strcat("D" + num2str(iii
+ 1) + ":G" + num2str(iii + 1)));
end
```

ПРИЛОЖЕНИЕ Б

КОД MATLAB ДЛЯ ОБРАБОТКИ ИЗОБРАЖЕНИЙ И ЗАПУСКА АЛГОРИТМОВ ПОИСКА (ФАЙЛ IMAGEPROCESSING.M)

```
close all;
clear all;
clc;
% параметры устройства
d = 0.0003;      % диаметр отверстия
t = 0.00005;      % толщина отверстия
h = 0.0007;       % высота отверстия
format long
height = 4.51e-3;     % Размеры матрицы
width = 2.88e-3;
pxW = 752;
pxH = 480;
pxSize = width / pxW;
angle = deg2rad(0 : 359);            % вспомогательный массив углов
defCircleX = (d/2) * cos(angle);              % координата x контура пятна
defCircleY = (d/2) * sin(angle);              % координата y контура пятна
xPx = ceil((defCircleX + width/2) / pxSize);  % x - контур пятна в пикселях
yPx = ceil((defCircleY + height/2) / pxSize);  % y - контур пятна в пикселях
defPxRadius = ceil((max(xPx) - min(xPx)) / 2);
filename = 'compare.xlsx';
for iii = 1 : 1000
    % чтение картинки и перевод в ч-б
    [image, colorMap] = imread(strcat("./Images/circle_", num2str(iii),
".png"));
    imageGS = rgb2gray(image);
    % figure, imshow(image);
    % figure, imshow(imageGS);
    [rows, columns, numberOfColorChannels] = size(image);
    x = 1:columns;
    y = 1:rows;
    imageProcessed = zeros(rows, columns, 3);
    %% поиск цветных пикселей перебором всех пикселей матрицы
    [brutX, brutY] = brutForce(imageGS);
    %% поиск рандомом
    [randX, randY] = randPick(imageGS);

    fileData = {brutY, brutX, randY, randX};
%     writecell(fileData, filename, 'sheet', 1, 'Range', strcat("H" + iii + 1));
    writecell(fileData, filename, 'sheet', 1, 'Range', strcat("H" + num2str(iii
+ 1) + ":K" + num2str(iii + 1)));
    strcat("H" + num2str(iii + 1) + ":K" + num2str(iii + 1))

end
```

ПРИЛОЖЕНИЕ В

АЛГОРИТМ ОБРАБОТКИ ИЗОБРАЖЕНИЯ RANDPICK.M

```
function [randX, randY] = randPick(imageGS)
        % параметры устройства
    d = 0.0003;      % диаметр отверстия
    t = 0.00005;      % толщина отверстия
    h = 0.0007;       % высота отверстия

    format long
    height = 4.51e-3;     % Размеры матрицы
    width = 2.88e-3;
    pxW = 752;
    pxH = 480;
    pxSize = width / pxW;

    angle = deg2rad(0 : 359);           % вспомогательный массив углов
    defCircleX = (d/2) * cos(angle);              % координата x контура пятна
    defCircleY = (d/2) * sin(angle);              % координата y контура пятна
    xPx = ceil((defCircleX + width/2) / pxSize);  % x - контур пятна в пикселях
    yPx = ceil((defCircleY + height/2) / pxSize);  % y - контур пятна в пикселях
    defPxRadius = ceil((max(xPx) - min(xPx)) / 2);

    columns = 752;
    rows = 480;
    x = 1:columns;
    y = 1:rows;

    coloredX2 = [];
    coloredY2 = [];

    currentX = 0;
    currentY = 0;

    find = false;
    repeat = zeros(pxH, pxW);

    % поиск первой точки пятна
    iter = 1;
    while(find == 0)
        if (iter > 10000)
            break;
        end
        randX = randi(columns, 1);
        randY = randi(rows, 1);

        if (repeat(randY, randX) == 1)
            if (any(repeat(:) == 0))
            continue;
            else break;
            end
        end

        repeat(randY, randX) = 1;

        if(squeeze(imageGS(randY, randX, :)) ~= 0)
            coloredX2(end + 1) = randX;
            coloredY2(end + 1) = randY;
            find = true;
        end
```

```
    iter = iter + 1;
end

if (length(coloredX2) == 0 || length(coloredY2) == 0)
    randX = 0;
    ranY = 0;
    return;
end
% Т.К. ПОЛУЧЕННОЕ ПЯТНО 100% МЕНЬШЕ ЧЕМ ТО, ЧТО ПОЛУЧАЕТСЯ БЕЗ НАКЛОНА
% МАТРИЦЫ МОЖНО ПОИСКАТЬ ПИКСЕЛИ В ПРЕДЕЛАХ ДИАМЕТРА НЕИЗМЕННЁНОГО КРУГА
% В 4 КВАДРАНТАХ

% левый верхний квадрат граница
leftTopX = coloredX2(1) - defPxRadius *  2;
leftTopY = coloredY2(1) - defPxRadius * 2;
% левый нижний квадрат граница
leftBottomX = coloredX2(1) - defPxRadius * 2;
leftBottomY = coloredY2(1) + defPxRadius * 2;
% правый верхний квадрат граница
rightTopX = coloredX2(1) + defPxRadius * 2;
rightTopY = coloredY2(1) - defPxRadius * 2;
% правый нижний квадрат
rightBottomX = coloredX2(1) + defPxRadius * 2;
rightBottomY = coloredY2(1) + defPxRadius * 2;

quadroPositionsX = [leftTopX, rightTopX, rightBottomX, leftBottomX];
quadroPositionsY = [leftTopY, rightTopY, rightBottomY, leftBottomY];

% проверка не выходят ли координаты за границы матрицы, если выходят ставим
% = размер матрицы или 0
for i = 1 : 4
   if (quadroPositionsX(i) <= 0)
      quadroPositionsX(i) = 1;
   else
       if (quadroPositionsX(i) > pxW)
           quadroPositionsX(i) = pxW;
       end
   end
   if (quadroPositionsY(i) <= 0)
      quadroPositionsY(i) = 1;
   else
       if (quadroPositionsY(i) > pxH)
           quadroPositionsY(i) = pxH;
       end
   end
end

isFlag = false;
isFlag2 = false;

for i = quadroPositionsX(1) : quadroPositionsX(3)
   for j = quadroPositionsY(1) : quadroPositionsY(3)
       if (i <= 0 || j <= 0)
          continue;
       end
        if(squeeze(imageGS(j, i, :)) ~= 0)
            coloredX2(end + 1) = i;
            coloredY2(end + 1) = j;
            isFlag2 = true;
            if (length(coloredX2) == 2)
               isFlag = true;
            end
        end
```

```
        end
        if (isFlag)
            if (~isFlag2)
              break;
            end
        end
    end

    randX = ceil(sum(coloredX2) / length(coloredX2));
    randY = ceil(sum(coloredY2) / length(coloredY2));
end
```

ПРИЛОЖЕНИЕ Г

АЛГОРИТМ ОБРАБОТКИ ИЗОБРАЖЕНИЯ BRUTFORCE.M

```
function [brutX, brutY] = brutForce(imageGS)
    % параметры устройства
    d = 0.0003;      % диаметр отверстия
    t = 0.00005;      % толщина отверстия
    h = 0.0007;       % высота отверстия

    format long
    height = 4.51e-3;     % Размеры матрицы
    width = 2.88e-3;
    pxW = 752;
    pxH = 480;
    pxSize = width / pxW;

    angle = deg2rad(0 : 359);            % вспомогательный массив углов
    defCircleX = (d/2) * cos(angle);              % координата x контура пятна
    defCircleY = (d/2) * sin(angle);              % координата y контура пятна
    xPx = ceil((defCircleX + width/2) / pxSize);  % x - контур пятна в пикселях
    yPx = ceil((defCircleY + height/2) / pxSize);  % y - контур пятна в пикселях
    defPxRadius = ceil((max(xPx) - min(xPx)) / 2);

    columns = 752;
    rows = 480;
    x = 1:columns;
    y = 1:rows;

    coloredX = [];
    coloredY = [];

    isFlag = false;
    isFlag2 = false;

    for  i = 1 : columns
        for j = 1 : rows
            if(squeeze(imageGS(j, i, :)) ~= 0)
                coloredX(end + 1) = i;
                coloredY(end + 1) = j;
                isFlag2 = true;
                if (length(coloredX) == 1)
                   isFlag = true;
                end
            end
        end
        if (isFlag)
            if (~isFlag2)
               break;
            end
        end
        isFlag2 = false;
    end

    if (length(coloredX) == 0 || length(coloredY) == 0)
       brutX = 0;
       brutY = 0;
       return;
    end
        brutX = ceil(sum(coloredX) / length(coloredX));
        brutY = ceil(sum(coloredY) / length(coloredY));
end
```