# 2. Разработка комплексного метода обработки потоковых данных

В данной главе производится описание разработанного робастного адаптивного алгоритма распознавания выпуклых объектов в потоковых данных инвариантный к уровню как собственных шумов оптических сенсоров, так и внешних помех. Разработанный алгоритм не требователен к вычислительным ресурсам и предназначен для реализации на базе SOC (system on chip).

Входными данными разработанного алгоритма являются массив показаний датчиков определения расстояния. Данные датчики могут быть заменены, например, на датчики приближения.

На рисунке 1 представлен репрезентативный массив значений входных данных алгоритма, где по оси X отмечено положение десяти датчиков от a до j, а по оси Y измеренные построчно расстояния.

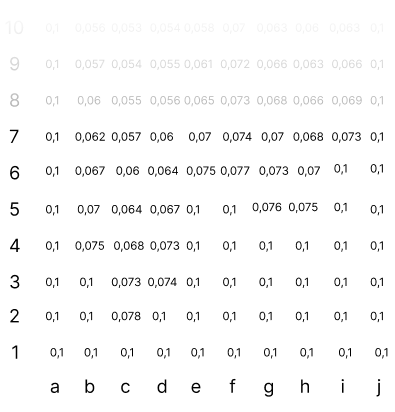


Рисунок 1 - Репрезентативный массив значений входных данных алгоритма

Примером датчика определения расстояния может служить VL53L0X, который работает на основе принципа лидара. Он использует инфракрасный лазер для измерения расстояния до объекта на основе времени, за которое отраженный лазерный сигнал возвращается обратно к датчику.

Набор сигналов от всех датчиков дает представление о контуре одного "среза" сканируемой области. Считывание нескольких срезов позволяет получить трехмерное представление о контуре сканируемой области.

Для решения поставленной задачи фильтрации данных необходимо учесть ряд свойств и выбрать наиболее подходящий для данного случая фильтр. Процесс фильтрации данных разделен на несколько этапов.

Первый этап – сбор данных. На данном этапе процесса фильтрации производится сбор и запись данных с датчиков для последующей обработки.

Далее следует этап предобработки данных. В процессе предобработки осуществляется фильтрация данных от возможных выбросов или шумов. Также учитываются физические ограничения, такие как предельные значения, и производится удаление данных с низкой интенсивностью. Рисунок 2 демонстрирует важность определения порога вхождения (ПВ) при фильтрации сигналов низкой интенсивности. Пороговый уровень играет ключевую роль в процессе фильтрации, позволяя выделять наиболее значимые сигналы и подавлять сигналы низкой интенсивности.

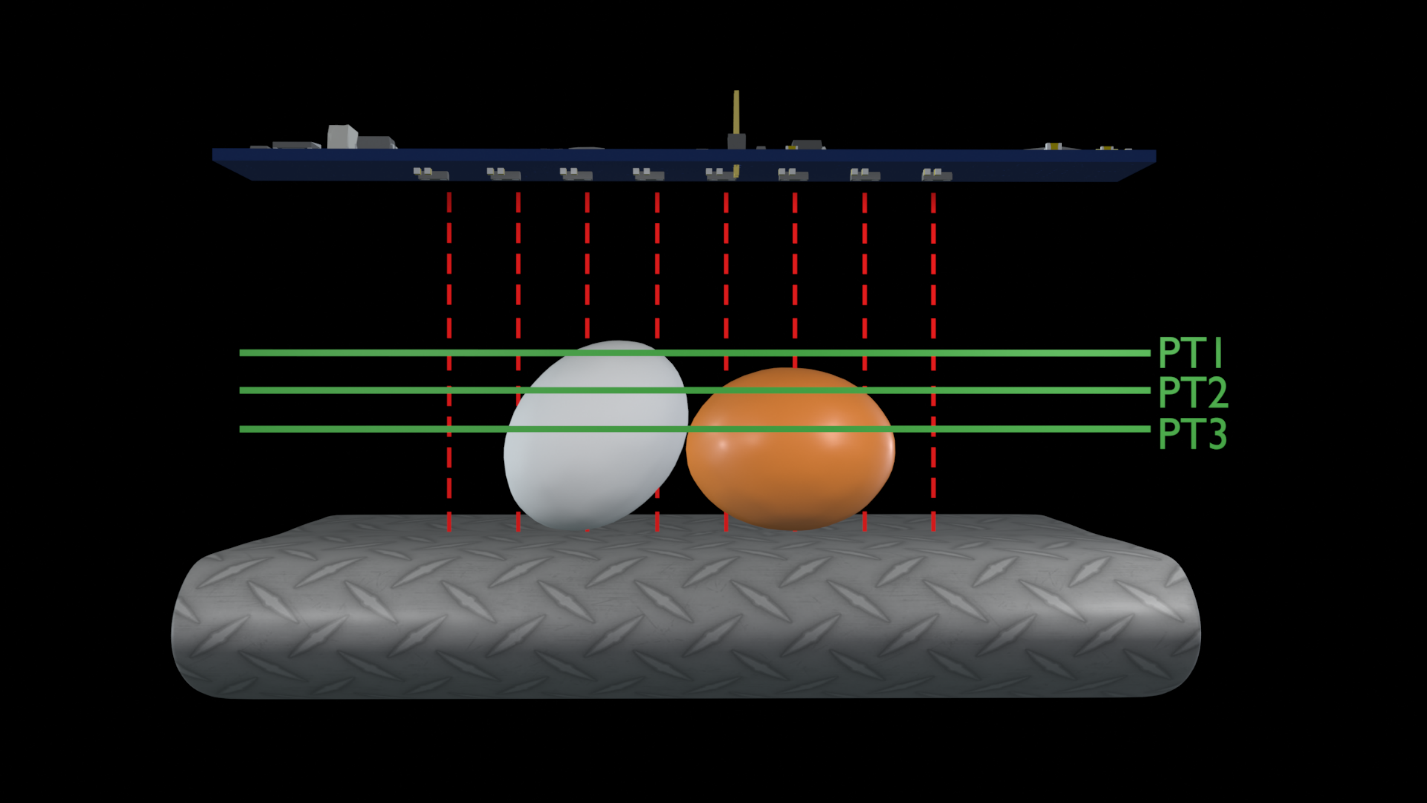


Рисунок 2 – Демонстрация важности ПВ

Алгоритм работы требует установления оптимального порога вхождения, который определяет, какие сигналы будут отфильтрованы, а какие останутся. Важно подобрать ПВ таким образом, чтобы он был достаточно чувствителен, чтобы обнаружить даже самые слабые сигналы или объекты, но при этом не был настолько низким, чтобы захватывать шумы.

Последний этап – применение фильтра. Важным шагом в процессе фильтрации является выбор конкретного фильтра, который будет применяться к данным. Целью применения фильтра является удаление шумов, выбросов или несовершенств оборудования.

На рисунке PT2 является показателем подходящего порога вхождения, так как он ориентирован на обнаружение сигналов, превышающих шумы или нерелевантные сигналы низкой интенсивности. При таком уровне порога отфильтровываются незначительные сигналы, но сохраняются важные детали, что важно для последующего анализа данных.

Теоретический порог вхождения на уровне PT1 был бы слишком низким, потому что при таком значении могли бы потеряться даже самые маленькие объекты или сигналы, которые необходимо обнаружить. С другой стороны, ПВ на уровне PT3 был бы слишком высоким, и такой порог не позволил бы обнаружить края или грани объектов, что также является важным аспектом при обработке данных.

Поэтому правильный выбор порога вхождения играет решающую роль в процессе фильтрации сигналов, помогая сохранить полезные данные и минимизировать воздействие шумов или нежелательных сигналов. Только оптимально подобранный порог обеспечивает эффективное и точное выделение значимых сигналов для дальнейших анализов или использования.

Последний этап – применение фильтра. Важным шагом в процессе фильтрации является выбор конкретного фильтра, который будет применяться к данным. Целью применения фильтра является удаление шумов, выбросов или несовершенств оборудования.

При выборе оптимального метода фильтрации важно учитывать отношение сигнал/шум (SNR), так как оно является ключевым фактором, определяющим эффективность фильтра и его способность выделять и сохранять полезную информацию.

Высокое значение SNR (1) позволяет более эффективно фильтровать шум и сохранять качество сигнала. Выбор метода фильтрации должен учитывать такие параметры, как уровень шума в сигнале, характеристики сигнала и возможные искажения. Если SNR невысоко, может потребоваться более сложный метод фильтрации или дополнительные преобразования сигнала перед фильтрацией.

(2)

где - энергетический спектр неискаженного изображения;

- энергетический спектр неискаженного изображения.

При выборе оптимального метода фильтрации важно учитывать не только отношение сигнал/шум, но и специфические требования задачи, применимость метода к данному типу сигнала, вычислительную сложность алгоритма и другие параметры. Только учитывая все эти аспекты, можно выбрать наиболее подходящий и эффективный метод фильтрации сигнала с учетом его отношения к шуму.

Таким образом, для эффективной фильтрации данных с датчиков необходимо выбирать соответствующий фильтр, учитывая специфику сигнала и требования к итоговым данным. Подходящий выбор фильтра обеспечит эффективную фильтрацию шумов и артефактов, а также сохранение важных деталей сигнала, что в конечном итоге повысит качество и надежность результатов обработки сигналов.

При наличии шума или непостоянной скорости движения объектов, стандартные методы фильтрации могут давать неточные результаты.

Один из аналогов алгоритма скользящего окна с контролем может быть применен для решения подобных задач. Алгоритм, основываясь на предположении о непрерывности скорости движения объектов, позволяет более точно определить их скорость и направление движения для корректной работы алгоритма обнаружения объектов.

Алгоритм использует окно заданного размера n, которое перемещается по временному ряду данных. Размер окна n зависит от количества объектов и максимальной скорости, чем больше объекты, тем больше значение n. В каждом положении окна происходит анализ данных для обнаружения объектов, на основе этого происходит оценка их скорости, что определяет размер окна m (2). При заполнении окна m данными вычисляется среднее арифметическое всех столбцов, которое затем используется для обработки алгоритмом поиска объектов.

*(2)*

где j – количество объектов, на основе которых производятся расчеты;

xi – количество детектированных показателей i-го объекта;

k – коэффициент пропорциональности, устанавливается экспериментально по результатам анализа соответствия формы объектов заданному образцу.

Основное преимущество данного алгоритма состоит в том, что он учитывает изменение скорости движения объектов во времени и способен дать более точные результаты при наличии шума или непостоянной скорости. Схема работы изображен на рисунке 5.

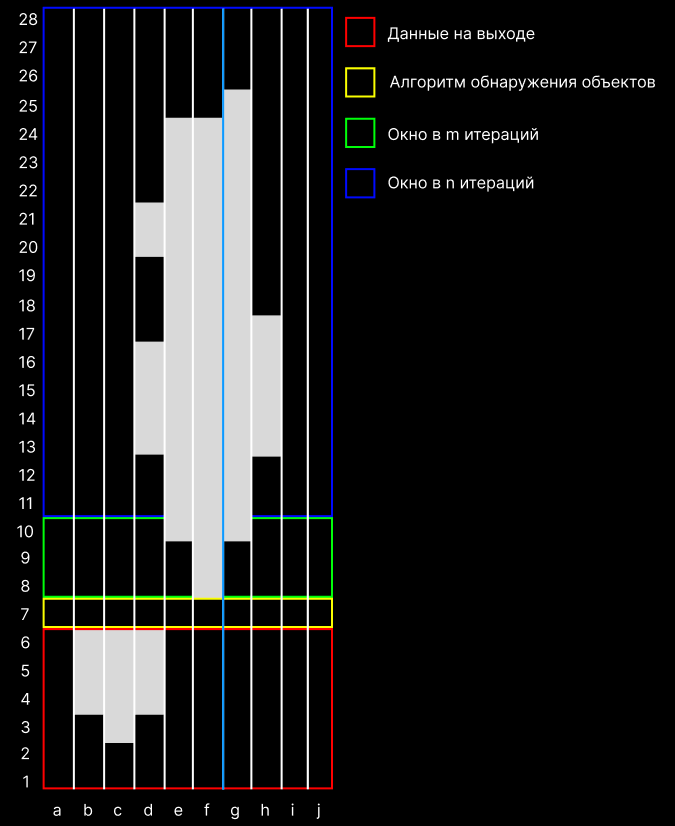


Рисунок 5 – Схема работы алгоритма классификации и скользящего окна

Блок-схема алгоритма представлена на рисунке 6.

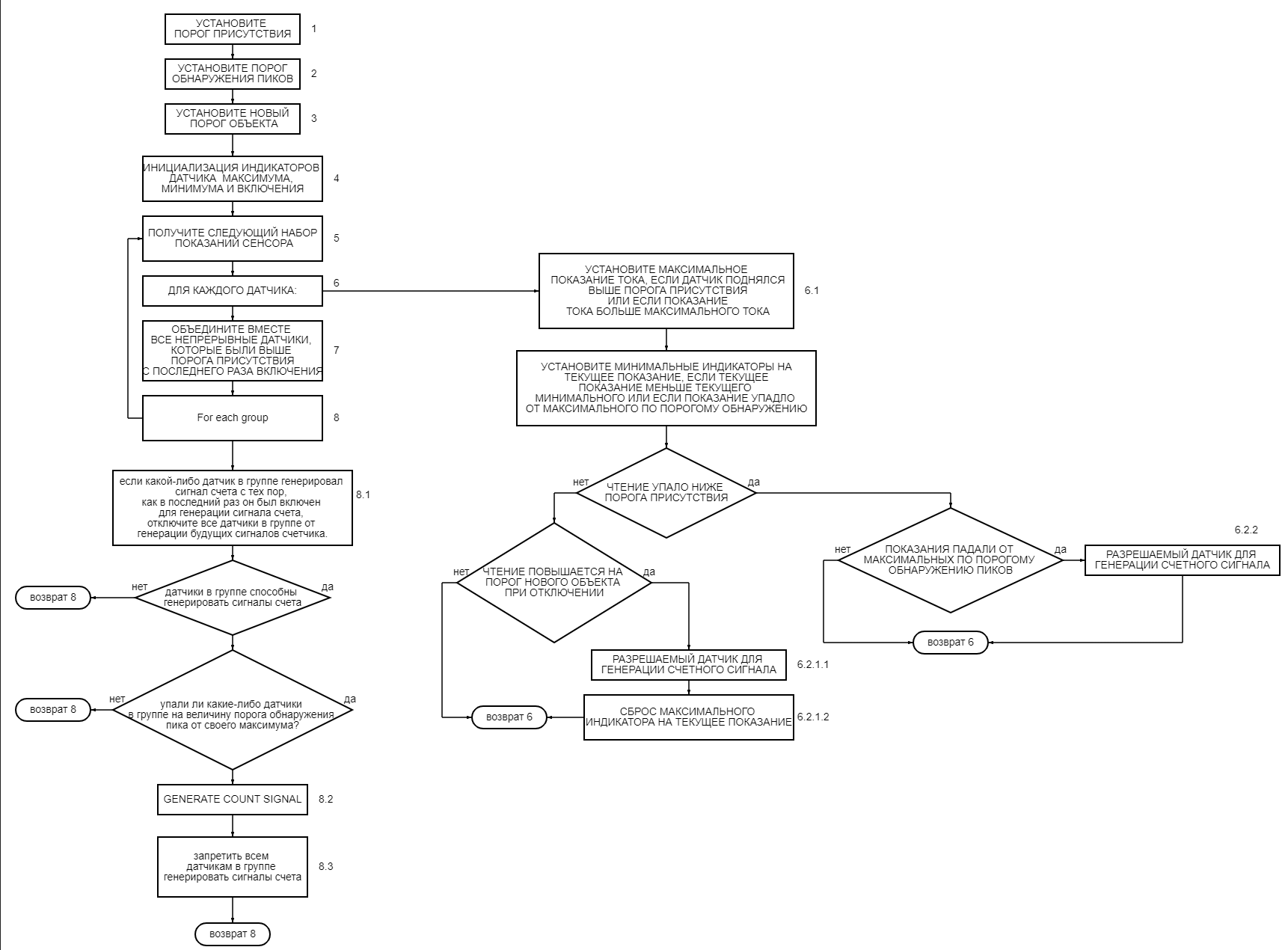


Рисунок 6 – Блок-схема алгоритма

Основной принцип работы алгоритма детектирования объектов выпуклой формы заключается в создании группы вокруг объекта, добавление датчиков в существующую группу, и поиск вершины путем выбора тройки датчиков с наименьшей суммой и определения центра этой тройки. Также алгоритм включает в себя отключение от группы на основе обнаружения другого объекта, корректировку принадлежности датчиков к группам путем проверки экстремумов, и отключение датчиков при прохождении объектов.

Блок-схема алгоритма представлена на рисунке 3.

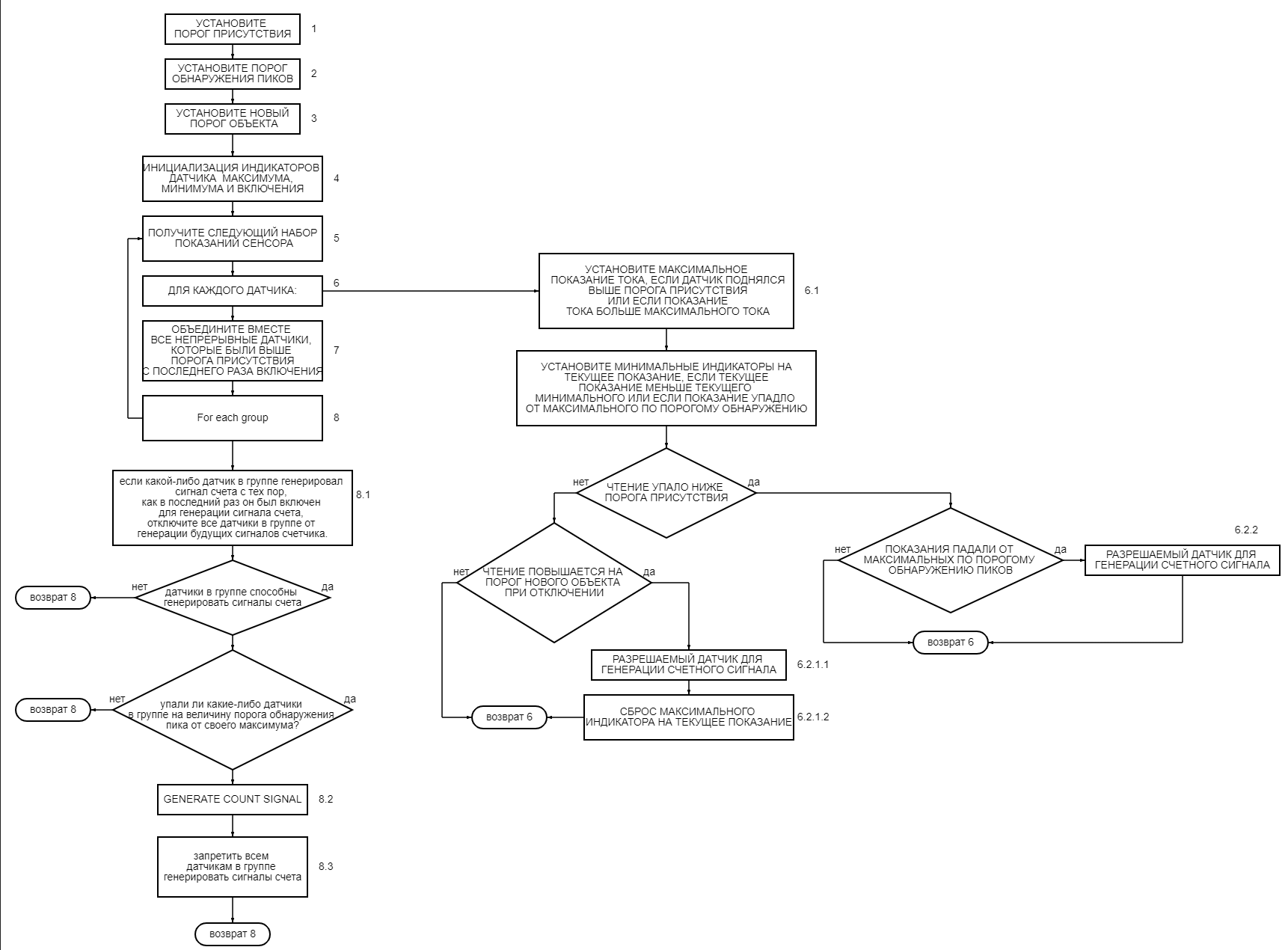


Рисунок 3 – Блок-схема алгоритма

Ниже приводится краткое пошаговое описание алгоритма в отношении показаний матрицы сигналов, показанных на рисунке 4.

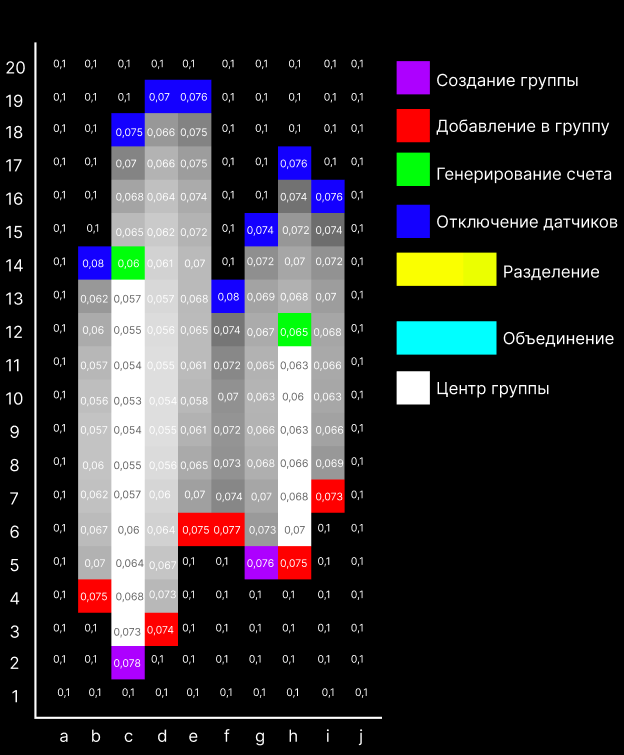


Рисунок 4 - Показания матрицы сигналов.

Показание 1. Ни одно показание не превысило, все датчики остаются выключенными.

Показание 2. Показание датчика c превышает ПВ. Так как по соседству отсутствуют созданные группы, то создается новая группа (группа №1). Центр данной группы устанавливается на датчик c.

Показание 3. Показание датчика d превышает ПВ. Так как по соседству существует группа, то датчик d добавляется в данную группу (группа №1). Центр данной группы устанавливается на датчик c.

Показание 4. Показание датчика b превышает ПВ. Так как по соседству существует группа, то датчик b добавляется в данную группу (группа №1). Центр данной группы устанавливается на датчик c.

Показание 5. Показание датчика g превышает ПВ. Так как по соседству отсутствуют созданные группы, то создается новая группа (группа №2). Показание датчика h превышает ПВ. Так как по соседству существует группа, то датчик h добавляется в данную группу (группа №2). Центр группы 1 устанавливается на датчик с, а центр группы 2 устанавливается на на датчик h.

Показание 6. Показание датчика e превышает ПВ. Так как по соседству существует группа, то датчик e добавляется в группу №1. Показание датчика f превышает ПВ. Так как рядом находятся 2 группы, то датчик присоединятся к той, где показатель соседнего датчика меньше. Центр группы 1 устанавливается на датчик с, а центр группы 2 устанавливается на датчик h.

Показание 7. Показание датчика i превышает ПВ. Так как рядом находится группа №2, то датчик присоединяется к ней. Центр группы 1 устанавливается на датчик с, а центр группы 2 устанавливается на на датчик h.

Показание 8-11. Центр группы 1 устанавливается на датчик с, а центр группы 2 устанавливается на на датчик h.

Показание 12. Значение тройки датчиков с наименьшей суммой группы 2 упало больше, чем на 5%. Это означает, что найдена вершина объекта и генерируется счет. Всем датчикам в этой группе устанавливается флаг, который запрещает генерировать счет. Центр группы 1 устанавливается на датчик с.

Показание 13. Показание датчика f упало ниже ПВ. Данный датчик отключается. Центр группы 1 устанавливается на датчик с.

Показание 14. Значение тройки датчиков с наименьшей суммой группы 2 упало больше, чем на 5%. Это означает, что найдена вершина объекта и генерируется счет. Всем датчикам в этой группе устанавливается флаг, который запрещает генерировать счет. Показание датчика b упало ниже ПВ. Данный датчик отключается.

Показание 15. Показание датчика g упало ниже ПВ. Данный датчик отключается.

Показание 16. Показание датчика i упало ниже ПВ. Данный датчик отключается.

Показание 17. Показание датчика h упало ниже ПВ. Данный датчик отключается.

Показание 18. Показание датчика c упало ниже ПВ. Данный датчик отключается.

Показание 19. Показание датчиков d, e упали ниже ПВ. Данные датчики отключаются.

Описанный во втором разделе адаптивный алгоритм предоставляет собой эффективное и надежное решение для обработки непрерывных данных в реальном времени.