# 3. Экспериментальная проверка разработанного алгоритма

# 3.1 Организация эксперимента

# 3.1.1 Проектирование и разработка модели в Unity

Реализация алгоритма включает использование датчиков изменения высоты, которые располагаются на определенном расстоянии друг от друга. Сигналы от этих датчиков передаются на центральный компьютер или контроллер для обработки данных.

Для эмуляции функциональности устройства был выбран игровой движок Unity, который является популярным и предоставляет возможность создания виртуальных сред и моделирования физических процессов. В рамках Unity была разработана симуляция, которая имитировала считывание показателей с датчиков. Таким образом, была осуществлена проверка алгоритма определения количества яиц без использования реальных яиц и физической модели. Геометрические показатели зависимости размера яиц от категории соответствует рисунку 1.

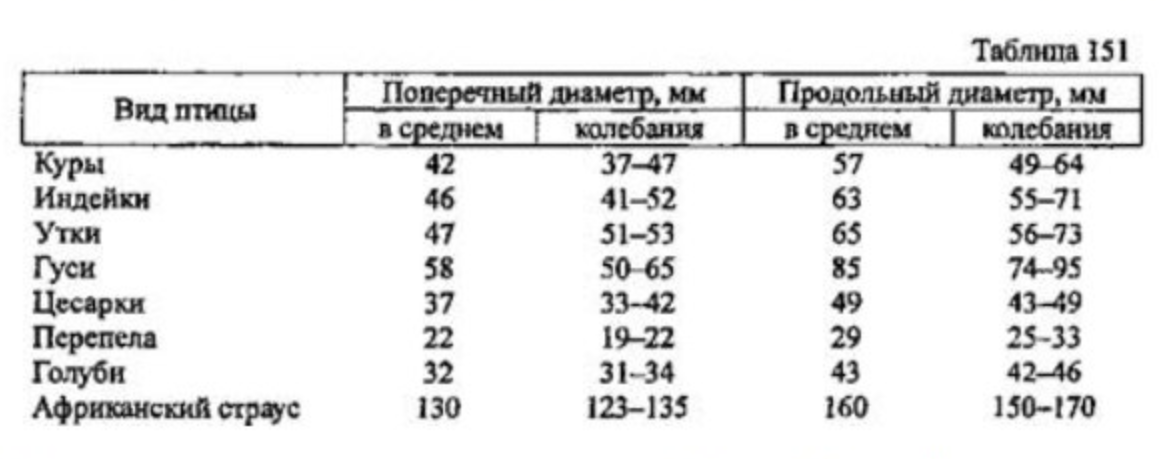


Рисунок 1 – Геометрические показатели зависимости размера яиц от категории

На рисунке 2 представлена генерация объектов, учитывая различные категории и положение яиц.

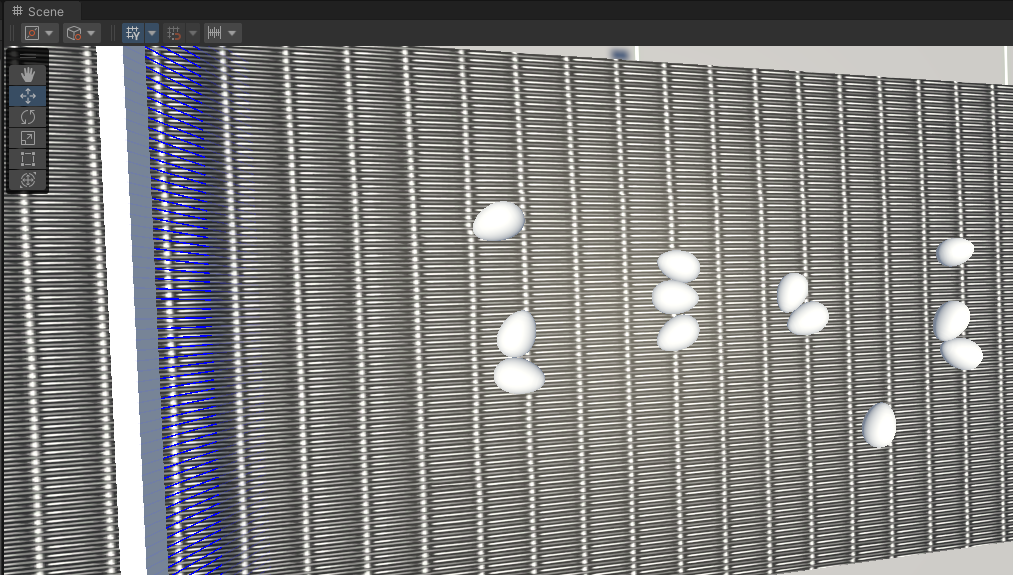


Рисунок 2 – Генерация яиц

Код генерации объектов представлен листингом 1.

Листинг 1 - Код генерации объектов

void Update()

{

float width = ConvWidthScript.getWidth();

if (Time.time > nextSpawn && GlobalVar.GetIsSpawn())

{

nextSpawn = Time.time + spawnDelay;

CalcEgg.addEggSpawnCol(spawnCol);

for (int i = 0; i < spawnCol; i++)

{

randomX = 0;

float rotateEgg = Random.Range(0.0f, 360.0f);

whereToSpawn = new Vector3(randomX, 2.0f, -7f);

GameObject Enemy = Instantiate(go, whereToSpawn, Quaternion.identity);

Enemy.transform.Rotate(0.0f, rotateEgg, 90f);

Enemy.transform.localScale = new Vector3(0.011f \* randomScale, 0.011f \* randomScale, 0.011f \* randomScale);//0.021

}

}

}

В Unity роль датчиков выполняют примитивные кубы, которые отправляют лучи (ray) через определенные промежутки времени. Основные характеристики шума и частоты обновления регулируются на основе настоящих датчиков VL53L0X. Внешний вид датчиков в Unity отображен на рисунке 3. Ширина датчиков, частота обновления и расстояние до конвейера могут быть настроены. Скорость движения конвейера можно задать в пределах от 1м/мин до 12 м/мин.

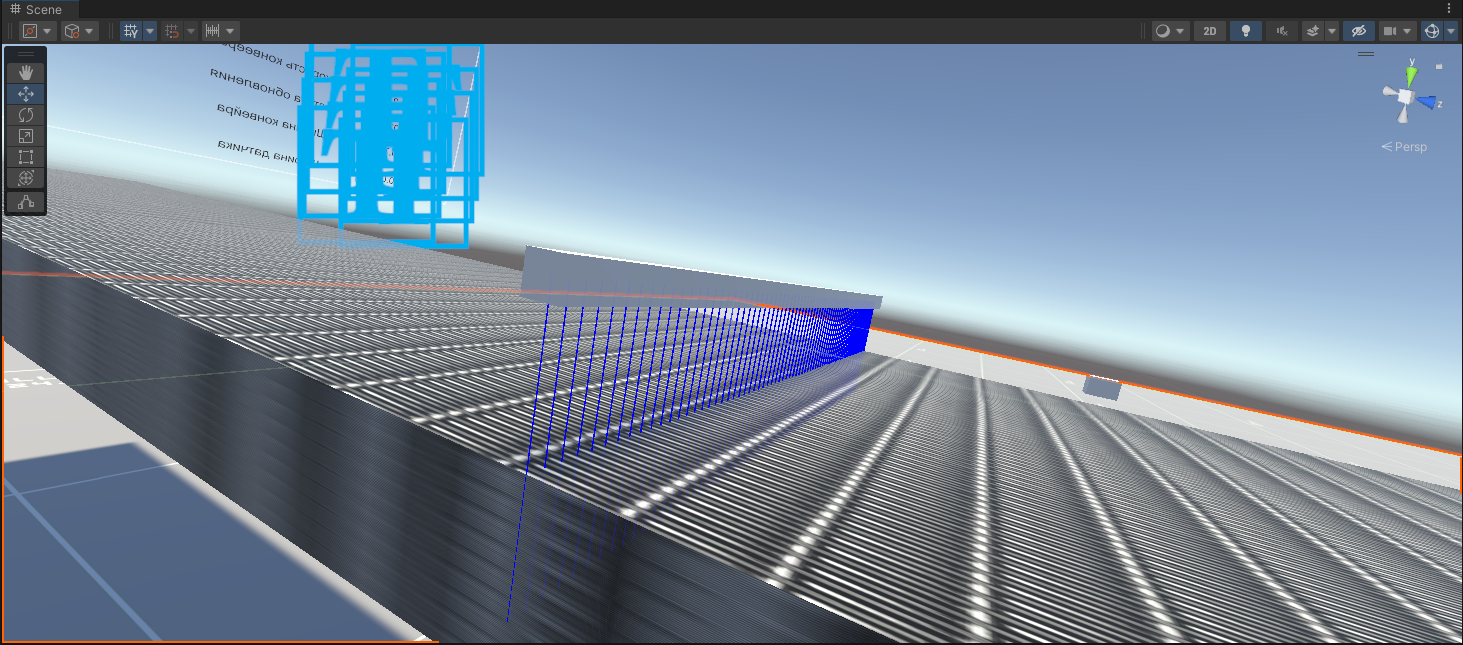


Рисунок 3 – Внешний вид датчиков

Код фиксирования показателей датчика представлен листингом 2.

Листинг 2 - Код фиксирования показателей датчика

void FixedUpdate()

{

nextUpdate = Time.time + GlobalVar.getSensorUpdateDelay();

Ray ray = new Ray(transform.position, transform.forward \* 0.15f);

Debug.DrawRay(transform.position, transform.forward \* 0.15f, Color.blue);

RaycastHit hit;

if (!Physics.Raycast(ray, out hit, 0.45f))

return;

float randomScale = Random.Range(-GlobalVar.getNoise(), GlobalVar.getNoise());

mas.Add(hit.distance + randomScale);

bool isIndexLast = (index == HeightMap.getMass().Count - 1);

if (isIndexLast)

{

HeightMap.CheckSensors();

}

if (hit.collider.gameObject.GetInstanceID() == -1268)

return;

float val = mas[mas.Count - 1];

if (val >= GlobalVar.getMinDistance())

return;

hit.collider.gameObject.GetComponent<Renderer>().material.color = Color.red;

Destroy(hit.collider.gameObject, 5f);

}

Результат генерации представлен на рисунке N.



Рисунок N -результат генерации

Записанные показания с датчиков сохраняются в бинарный файл. Структура бинарного файла представлена на рисунке 4.

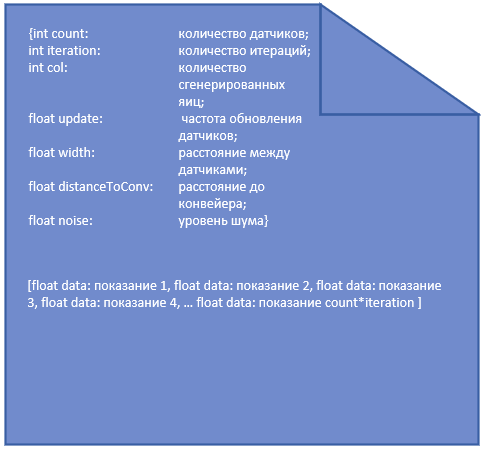


Рисунок 4 – Структура бинарного файла

Общий вид сцены представлен на рисунке 5.

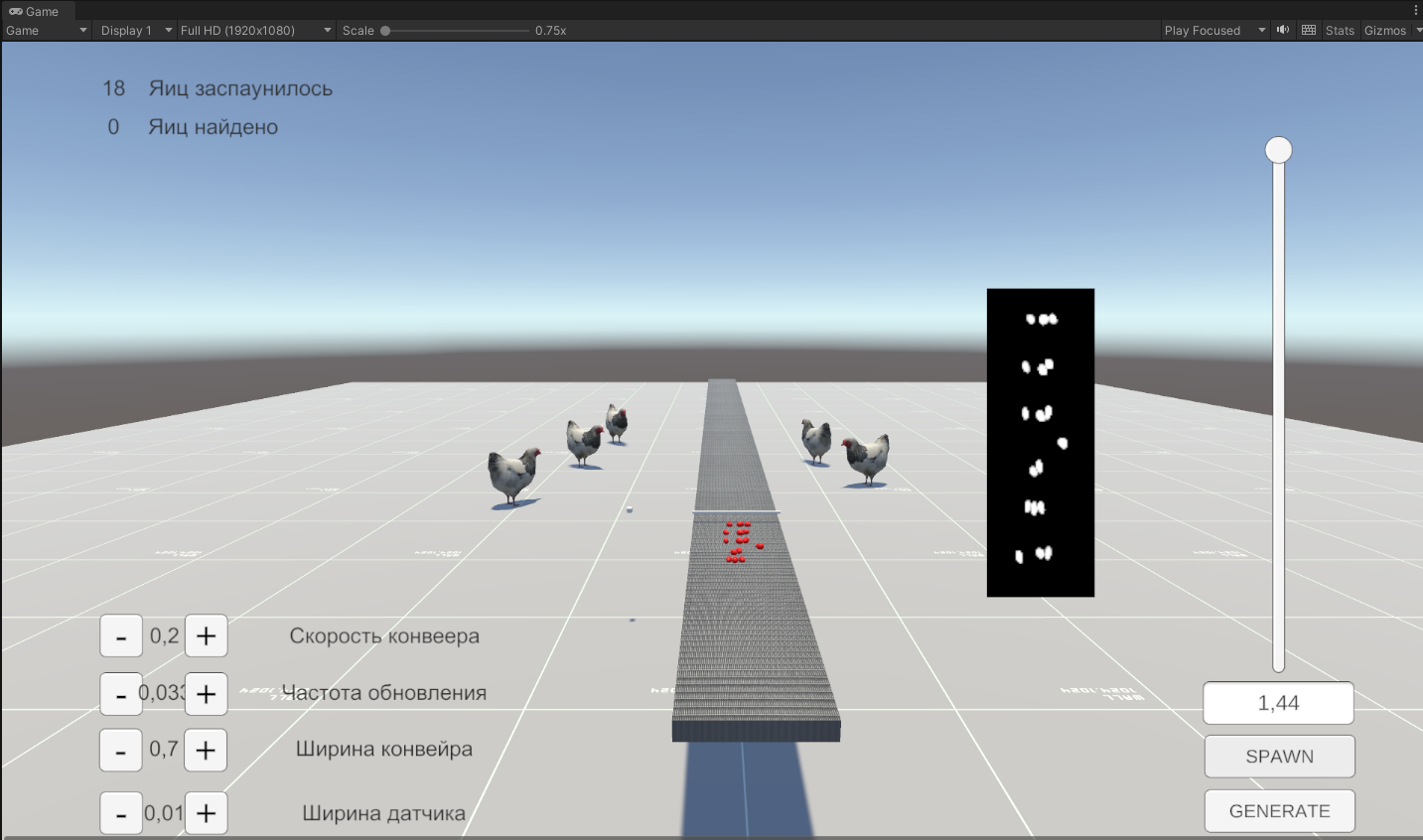


Рисунок 5 – Общий вид сцены

# 3.1.2 Проектирование и разработка приложения для оценки работы методов фильтрации

Приложение предназначено для оценки метода фильтрации. Блок-схема работы приложения представлена на рисунке 6.

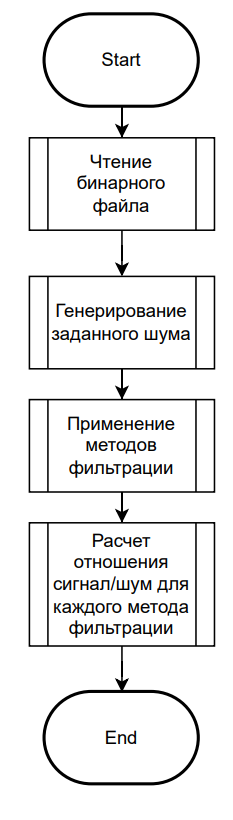


Рисунок 6 – Блок-схема работы приложения

Отношение сигнал/шум дает меру уровня информационного отношения мощности сигнала (т. е. исходного, неискаженного изображения) к уровню мощности шума. Изображения с низким шумом будут иметь большое значение SNR (2), а те же изображения с высоким уровнем шума — малое значение SNR. Само по себе данное соотношение имеет ограниченное значение, но оно является важной метрикой, используемой для описания характеристик алгоритмов восстановления.

(2)

где - энергетический спектр неискаженного изображения;

- энергетический спектр неискаженного изображения.

На основе результатов работы приложения по оценке пикового отношения, был выбран метод фильтрации данных - скользящее среднее с весом 16. На рисунке 8 представлена матица весов. Полученные результаты показали, что этот метод обеспечивает достаточное снижение уровня шума и сохранение структуры данных на карте вершин с яйцами. Пример работы программы представлен не рисунке 6.

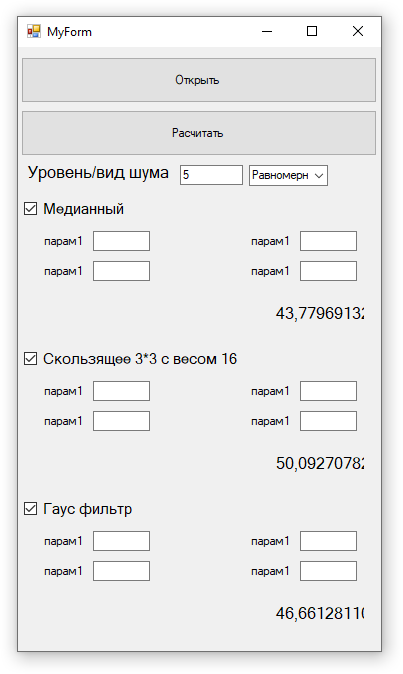


Рисунок 6 – результат работы приложения по оценке пикового отношения

Также следует отметить, что для каждой предметной области необходимо использовать свой метод фильтрации, чтобы обеспечить наилучший результат и оптимальное пиковое отношение сигнала к шуму в каждом конкретном случае.

# 3.1.3 Проектирование и разработка приложения для оценки работы алгоритма

Приложение предназначено для оценки работы алгоритма с заданным уровнем шума. Блок-схема работы приложения представлена на рисунке 7.



Рисунок 7 – Блок-схема работы приложения

Приложение читает бинарный файл, структура которого представлена на рисунке 4. Для генерации шума было использовано равномерное распределение. Общий вид приложения представлен на рисунке 5.

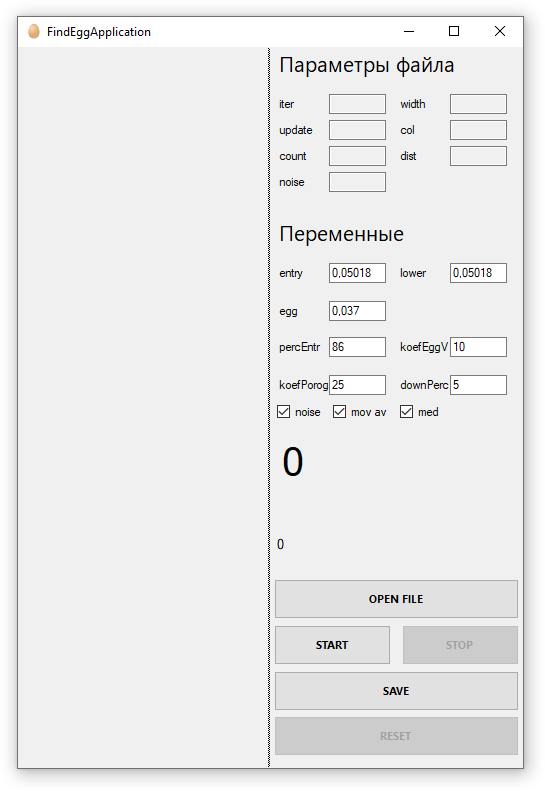


Рисунок 5 – Общий вид приложения

В блоке "Параметры файла" выводится общая информация из бинарного файла. пример открытого файла представлен на рисунке 6.

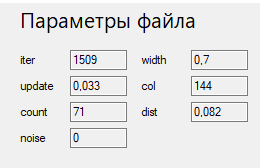


Рисунок 6 – Общая информация из бинарного файла

В блоке "Переменные" отображается порог обнаружения и переменные для работы алгоритма:

* entry - порог обнаружения;
* egg - базовый размер яйца, на основе которого будет рассчитываться порог обнаружения;
* percEntr – процент от базового размера яйца для расчета порога обнаружения;
* koefEggV - количество объектов, на основе которых производятся расчеты по формуле (1);
* koefPorog - коэффициент пропорциональности по формуле (1).

Для визуального отображения работы алгоритма, приложение генерирует изображение, на котором отображены основные его части по промеру с рисунка N(раздел 2). Пример изображения представлен на рисунке N.

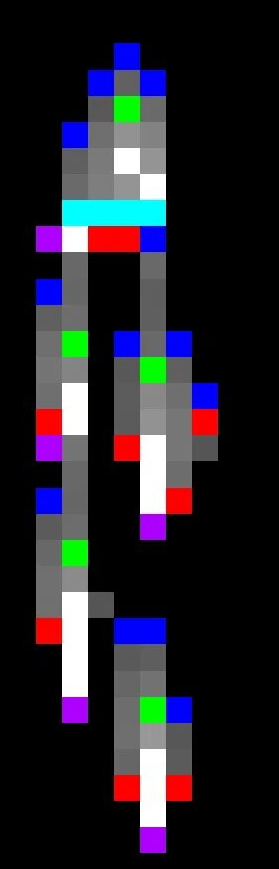


Рисунок N - Визуальное отображение работы алгоритма

Пример работы программы представлен на рисунке 7.

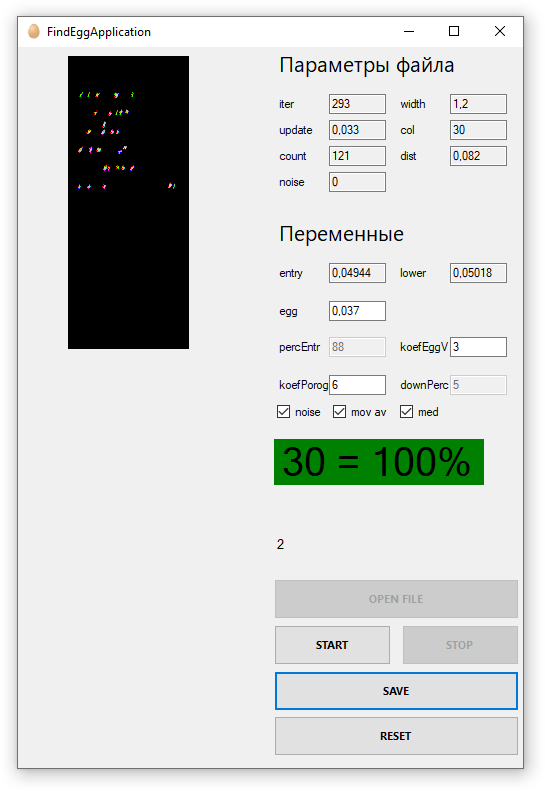


Рисунок 7 – Пример работы программы

# 3.2 Анализ результатов эксперимента

Набор из 500 яиц был пропущен через программу 50 раз для каждого уровня шума. Ячейки таблицы содержат проценты точности каждой итерации для каждого значения амплитуды шума. В строке total отображается средний уровень точности в %. Для расстояния между датчиками 1см оптимальный уровень шума = 6-7% и ниже. Таблица с результатом представлена в Приложении А.

# 3.3 Выводы по главе.

Для проведения экспериментальной проверки разработанного алгоритма была организована вычислительная модель в среде Unity. Были разработаны специальные приложения, позволяющие оценить работу методов фильтрации и алгоритма.

После проведения эксперимента был проведен анализ полученных результатов. Было выявлено, что разработанный алгоритм обладает определенными преимуществами по сравнению с существующими методами. Таким образом, эксперимент позволил подтвердить эффективность и применимость разработанного алгоритма в реальных условиях.

Эти результаты подтверждают актуальность и значимость проведенного исследования, а также показывают потенциал разработанного алгоритма в практических приложениях. В дальнейшем возможно его использование для улучшения качества обработки данных в различных областях, где требуется фильтрация сигналов.

# Приложение A

Результат работы алгоритма при расстоянии между датчиками 1,2 см

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Амплитуда шума в % от минимального размера яйца | | | | | | | | |
| N итерации | 3% | 4% | 5% | 6% | 7% | 8% | 9% | | 10% |
| 1 | 100 | 99,66666 | 100 | 100 | 100 | 100 | | 100 | 99,33334 |
| 2 | 100 | 100 | 100 | 100 | 100 | 100 | | 99,66666 | 99,33334 |
| 3 | 100 | 100 | 100 | 100 | 100 | 99,66666 | | 100 | 99 |
| 4 | 100 | 100 | 100 | 100 | 100 | 100 | | 99,66666 | 100 |
| 5 | 100 | 100 | 100 | 99,66666 | 100 | 100 | | 99,66666 | 99,66666 |
| 6 | 100 | 100 | 99,66666 | 100 | 100 | 99,66666 | | 99,33334 | 100 |
| 7 | 100 | 100 | 100 | 100 | 99,66666 | 99,33334 | | 99,33334 | 100 |
| 8 | 100 | 100 | 100 | 100 | 100 | 100 | | 100 | 99,66666 |
| 9 | 100 | 100 | 100 | 100 | 99,66666 | 100 | | 99,66666 | 99,66666 |
| 10 | 99,66666 | 100 | 100 | 99,66666 | 100 | 99,66666 | | 100 | 100 |
| 11 | 100 | 100 | 99,66666 | 99,66666 | 99,66666 | 100 | | 100 | 99,33334 |
| 12 | 100 | 100 | 100 | 100 | 99,66666 | 100 | | 100 | 100 |
| 13 | 100 | 100 | 99,66666 | 100 | 100 | 99,33334 | | 100 | 100 |
| 14 | 100 | 99,66666 | 100 | 99,66666 | 99,33334 | 99,66666 | | 99,33334 | 100 |
| 15 | 100 | 100 | 100 | 100 | 99,66666 | 99,66666 | | 100 | 99,66666 |
| 16 | 100 | 99,66666 | 100 | 99,66666 | 99,66666 | 99,66666 | | 99,66666 | 100 |
| 17 | 100 | 100 | 100 | 100 | 100 | 100 | | 99,66666 | 100 |
| 18 | 99,66666 | 100 | 100 | 100 | 100 | 100 | | 99,66666 | 99,66666 |
| 19 | 100 | 100 | 99,66666 | 99,66666 | 99,66666 | 100 | | 100 | 99,33334 |
| 20 | 100 | 100 | 100 | 100 | 100 | 100 | | 99,66666 | 99,66666 |
| 21 | 100 | 100 | 100 | 100 | 99,66666 | 99,66666 | | 99,66666 | 99,66666 |
| 22 | 100 | 100 | 100 | 100 | 100 | 99,66666 | | 100 | 100 |
| 23 | 100 | 100 | 100 | 99,66666 | 100 | 100 | | 100 | 99,66666 |
| 24 | 99,66666 | 99,66666 | 100 | 99,66666 | 99,66666 | 100 | | 100 | 100 |
| 25 | 100 | 100 | 99,66666 | 100 | 100 | 99,66666 | | 100 | 100 |
| 26 | 100 | 100 | 100 | 99,66666 | 100 | 100 | | 99 | 100 |
| 27 | 100 | 100 | 100 | 100 | 100 | 100 | | 100 | 99,66666 |
| 28 | 100 | 100 | 100 | 99,66666 | 100 | 99,66666 | | 100 | 99,66666 |
| 29 | 100 | 100 | 100 | 99,66666 | 100 | 100 | | 99,66666 | 99,66666 |
| 30 | 100 | 100 | 100 | 100 | 100 | 100 | | 99,66666 | 100 |
| 31 | 100 | 100 | 100 | 99,66666 | 100 | 99,66666 | | 100 | 100 |
| 32 | 100 | 100 | 100 | 100 | 100 | 100 | | 100 | 99,66666 |
| 33 | 100 | 100 | 100 | 100 | 100 | 99,66666 | | 100 | 100 |
| 34 | 100 | 100 | 100 | 100 | 99,66666 | 100 | | 100 | 99,33334 |
| 35 | 100 | 100 | 100 | 100 | 99,66666 | 99,33334 | | 100 | 100 |
| 36 | 100 | 100 | 100 | 100 | 99,66666 | 100 | | 100 | 99,33334 |
| 37 | 100 | 100 | 99,66666 | 100 | 99,66666 | 100 | | 99,66666 | 99,66666 |
| 38 | 100 | 100 | 100 | 99,66666 | 100 | 99,66666 | | 100 | 100 |
| 39 | 100 | 99,66666 | 99,66666 | 100 | 99,66666 | 99,66666 | | 99,66666 | 100 |
| 40 | 100 | 100 | 100 | 100 | 100 | 100 | | 99,66666 | 99,66666 |
| 41 | 100 | 100 | 100 | 100 | 100 | 100 | | 100 | 99,66666 |
| 42 | 100 | 100 | 100 | 99,66666 | 100 | 99,66666 | | 99,66666 | 100 |
| 43 | 100 | 100 | 99,66666 | 100 | 99,66666 | 99,66666 | | 100 | 99,66666 |
| 44 | 100 | 100 | 100 | 100 | 100 | 100 | | 100 | 100 |
| 45 | 100 | 100 | 100 | 100 | 99,66666 | 99,66666 | | 99,33334 | 99,66666 |
| 46 | 100 | 100 | 100 | 100 | 100 | 99,33334 | | 100 | 100 |
| 47 | 100 | 100 | 100 | 100 | 100 | 100 | | 99,66666 | 100 |
| 48 | 100 | 100 | 100 | 100 | 100 | 100 | | 100 | 99,66666 |
| 49 | 100 | 100 | 100 | 100 | 99,66666 | 100 | | 99,66666 | 99,33334 |
| 50 | 100 | 100 | 99,66666 | 100 | 99,66666 | 99,33334 | | 100 | 100 |
| Total | 99,98 | 99,96667 | 99,94 | 99,90668 | 99,86666 | 99,82001 | | 99,81333 | 99,76667 |