

Реализация простого примера обработки цифрового сигнала на C#

Ульянов Егор Андреевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье рассматривается процесс реализации простого примера обработки цифрового сигнала в среде разработки Visual Studio. Описывается использование фильтра нижних частот, который удаляет высокочастотные компоненты сигнала. Целью данной статьи является, применение языка программирования C#, для простой обработки сигнала. Практическим результатом является разработанное приложение обработчик сигнала.

Ключевые слова: C#, Visual Studio, фильтр нижних частот, обработка цифрового сигнала

Implementation of a simple example of digital signal processing in C#

Ulianov Egor Andreevich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article discusses the process of implementing a simple example of digital signal processing in the Visual Studio development environment. The use of a low-pass filter that removes high-frequency components of the signal is described. The purpose of this article is to use the C# programming language for simple signal processing. The practical result is the developed signal handler application.

Keywords: C#, Visual Studio, low-pass filter, digital signal processing

1. Введение

1.1 Актуальность исследования

Актуальность данного исследования обусловлена несколькими факторами.

- Во-первых, цифровые сигналы играют важную роль в различных областях, таких как телекоммуникации, медицина, звукозапись, видеозапись и многие другие. Обработка цифровых сигналов является необходимой задачей в этих областях, поскольку она позволяет извлечь полезную информацию из сигнала, устранить шум и искажения, и улучшить качество сигнала.
- Во-вторых, в современном мире данные стали одним из наиболее ценных ресурсов. Обработка цифровых сигналов позволяет извлекать

ценную информацию из этих данных, что делает данную тему актуальной для ряда областей, особенно для машинного обучения, обработки естественного языка и других областей, где анализ данных является ключевой задачей.

Таким образом, реализация простого примера обработки цифрового сигнала на C# имеет большую практическую значимость и может найти применение в различных областях, где обработка цифровых сигналов играет важную роль.

1.2 Обзор исследований

В своей работе Н. Н. Додобоев, О. И. Кукарцева, Я. А. Тынченко рассмотрели вопросы появления различных языков программирования (в частности C#), определения особенностей этих языков, а также составления основных видов и классификаций языков программирования [1]. З. С. Магомадова рассмотрела языки программирования высокого уровня, особенности, недостатки и сложности в изучении, а также описала несколько легких алгоритмов [2]. В своей работе П. А. Садовников, А. Ю. Дроздов, Ю. Н. Фонин описали процесс разработки программы представляющую собой эффективную реализацию программных компонентов, наиболее часто используемых при решении задач цифровой обработки сигнала [3]. I.I. Markovich, E.E. Zavtur в статье представили экспериментальные результаты предложенных методов на моделях сигналов [4].

1.3 Цель исследования

Цель исследования – применяя возможности среды разработки Visual Studio и языка программирования C#, реализовать простой пример обработки цифрового сигнала. Реализация данного примера обработки поможет понять принципы обработки цифровых сигналов, а также даст возможность применять их на практике в других областях.

2. Материалы и методы

Для создания обработки будем использовать программное обеспечение Visual Studio, а также язык программирования C#.

Для проекта использовалась Windows Presentation Foundation (WPF), платформа пользовательского интерфейса для создания клиентских приложений для настольных систем. Платформа разработки WPF поддерживает широкий набор компонентов для разработки приложений, включая модель приложения, ресурсы, элементы управления, графику, макет, привязки данных, документы и безопасность. Эта платформа является частью платформы .NET.

3 Результаты и дискуссия

Цифровая обработка сигнала – это область науки, занимающаяся преобразованием аналоговых сигналов в цифровой формат и их последующей обработкой. Цифровая обработка сигнала широко используется в различных

областях, включая телекоммуникации, медицину, науку о материалах, автоматизацию и многие другие. Преобразование аналоговых сигналов в цифровую форму начинается с использования аналогово-цифрового преобразователя (ADC). ADC преобразует аналоговый сигнал в цифровой формат, имеющий дискретные значения во времени. Каждый дискретизированный отсчет представляет собой численное значение, которое может быть обработано на компьютере или другом устройстве. Далее, цифровой сигнал подвергается различным аналитическим и обработкам техникам, таким как фильтрация, ограничение диапазона, усиление, уменьшение шума, сжатие, и т.д. Эти техники применяются для извлечения полезной информации из сигнала или для подготовки сигнала к последующему использованию. Конечным результатом цифровой обработки сигнала является измененный, улучшенный, или анализированный цифровой сигнал, который может быть использован для различных приложений, таких как передача данных, распознавание речи, обработка изображений, медицинское диагностирование, и многие другие. Цифровая обработка сигнала применяется во многих областях. Некоторые из наиболее частых применений цифровой обработки сигнала включают:

- Телекоммуникации: цифровая обработка сигнала занимается обработкой и передачей аудио- и видеосигналов через различные сети и каналы связи.
- Медицина: цифровая обработка сигнала используется для анализа медицинских изображений, таких как рентген, КТ, МРТ, маммография, для диагностики и лечения заболеваний.
- Обработка изображений: цифровая обработка сигнала используется для обработки изображений, включая улучшение качества фотографий, распознавание лиц, анализ изображений, компьютерное зрение.
- Звукозапись и акустическая обработка: цифровая обработка сигнала применяется для записи звука и его оптимизации при монтаже в мультимедийных проектах, в производстве музыки и в других акустических проектах.
- Электроника: в электронике цифровая обработка сигнала используется в различных устройствах, таких как радио, телевизоры, мобильные устройства, компьютеры и микроконтроллеры.
- Геолокация: цифровая обработка сигнала используется для обработки и анализа данных GPS, а также для локализации и навигации в автомобилях и других устройствах.
- Управление и автоматизация: цифровая обработка сигнала используется для управления и контроля в различных процессах автоматизации, таких как контроль качества, вычислительные системы управления.

- Финансы: цифровая обработка сигнала используется для предсказания финансовых тенденций, управления рисками и других задач, связанных с анализом финансовых данных.

Процесс обработки цифрового сигнала включает следующие шаги:

- Чтение и запись сигнала. Сначала необходимо загрузить цифровой сигнал в программу из файла или другого источника. Затем, после обработки сигнала, его можно сохранить обратно в файл или передать в другую программу.
- Подготовка сигнала. Некоторые обработки сигнала могут требовать подготовительных шагов, таких как нормализация, удаление шума, приведение к единому формату и т.д.
- Фильтрация сигнала. Фильтрация сигнала позволяет удалить шумы, неполадки и другие нежелательные компоненты из сигнала, что позволяет улучшить его качество.
- Усиление сигнала. После обработки и фильтрации сигнала может потребоваться его усиление, чтобы достичь необходимого уровня звуковой мощности при проигрывании или передаче сигнала.

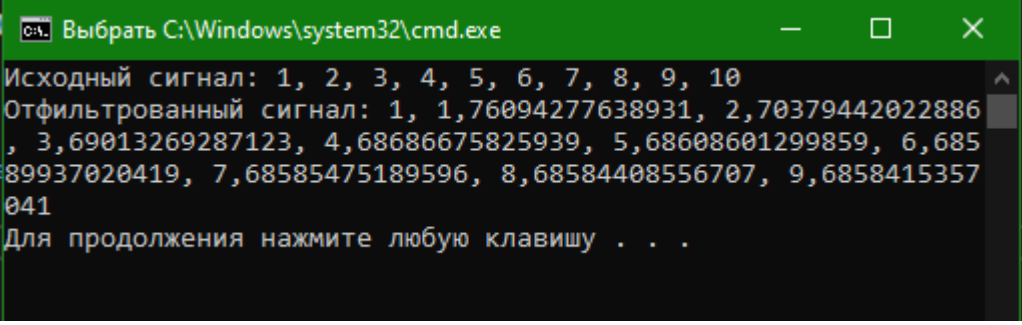
Эти шаги могут не применяться одновременно в каждой обработке сигнала, и могут быть опущены в зависимости от конкретной задачи. В данном случае будет использоваться фильтр нижних частот (Low-pass filter), который удаляет высокочастотные компоненты сигнала.

Перейдем к написанию кода, для фильтрации цифрового сигнала с заданными параметрами. Для начала создадим переменные: «samplingRate» типа «int» со значением 1000, которая задает частоту дискретизации для сигнала; переменные «fc» и «rc» типа «double» равную 50; и $1.0 / (2 * \text{Math.PI} * \text{fc})$ соответственно. «fc» задает частоту среза фильтра, а «rc» - константу времени фильтра ($1/2\pi RC$), где R - сопротивление и C - емкость. Далее создаем массив сигнала для фильтрации «double[] signal = new double[] { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }», и массив для результата фильтрации «double[] filteredSignal = new double[signal.Length]». Теперь напишем код для вычисления коэффициентов фильтра «double dt = 1.0 / samplingRate», «double alpha = rc / (rc + dt)», где вычисляются значения «dt» и «alpha». Значение «dt» - это интервал, между отдельными дискретными значениями, а значение «alpha» - это коэффициент, используемый для фильтрации. Применение фильтра происходит в цикле «for». Переменная у инициализируется первым значением в «signal». Затем для каждого элемента «signal[i]» фильтр рассчитывается на основе предыдущего значения «y», с использованием уравнения « $y = \text{alpha} * \text{signal}[i] + (1 - \text{alpha}) * y$ ». Результат записывается в filteredSignal[i]. Исходный сигнал и отфильтрованный сигнал выводятся в консоль с помощью метода «Console.WriteLine» и метода «String.Join», который объединяет элементы массива в строку, разделенную запятыми. В конце программы корректное отображение результата гарантирует метод «Console.ReadKey()», что ожидает нажатия пользователя для закрытия программы см. рисунок 1.

```
main.cs
1 using System;
2
3 class Program
4 {
5     static void Main(string[] args)
6     {
7         // Задаем частоту дискретизации
8         int samplingRate = 1000; // Герц
9
10        // Задаем параметры фильтра
11        double fc = 50; // Герц
12        double rc = 1.0 / (2 * Math.PI * fc);
13
14        // Создаем массив сигнала для фильтрации
15        double[] signal = new double[] { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
16
17        // Создаем массив для результата фильтрации
18        double[] filteredSignal = new double[signal.Length];
19
20        // Вычисляем коэффициенты фильтра
21        double dt = 1.0 / samplingRate;
22        double alpha = rc / (rc + dt);
23
24        // Применяем фильтр к сигналу
25        double y = signal[0];
26        for (int i = 0; i < signal.Length; i++)
27        {
28            y = alpha * signal[i] + (1 - alpha) * y;
29            filteredSignal[i] = y;
30        }
31
32        // Выводим результаты
33        Console.WriteLine("Исходный сигнал: " + String.Join(", ", signal));
34        Console.WriteLine("Отфильтрованный сигнал: " + String.Join(", ", filteredSignal));
35        Console.ReadKey();
36    }
37 }
```

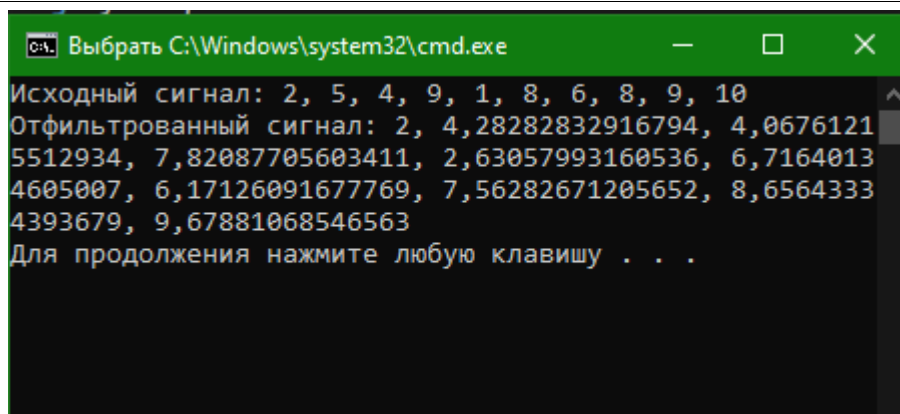
Рисунок 1 – Код программы

Переходим к тестированию программы см. рисунок 2-3.



```
C:\> Выбрать C:\Windows\system32\cmd.exe
Исходный сигнал: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
Отфильтрованный сигнал: 1, 1,76094277638931, 2,70379442022886
, 3,69013269287123, 4,68686675825939, 5,68608601299859, 6,685
89937020419, 7,68585475189596, 8,68584408556707, 9,6858415357
041
Для продолжения нажмите любую клавишу . . .
```

Рисунок 2 – Отображение исходного и отфильтрованного сигнала

A screenshot of a Windows command prompt window. The title bar is green and contains the text "Выбрать C:\Windows\system32\cmd.exe". The command prompt shows the following text:

```
Исходный сигнал: 2, 5, 4, 9, 1, 8, 6, 8, 9, 10
Отфильтрованный сигнал: 2, 4,28282832916794, 4,0676121
5512934, 7,82087705603411, 2,63057993160536, 6,7164013
4605007, 6,17126091677769, 7,56282671205652, 8,6564333
4393679, 9,67881068546563
Для продолжения нажмите любую клавишу . . .
```

Рисунок 1 – Проверка работы с другим исходным сигналом

4 Выводы

В рамках данного исследования описана реализация простого примера обработки цифрового сигнала в среде разработки Visual Studio. Данный пример является базовым и демонстрирует основные шаги создания и обработки цифрового сигнала в среде Visual Studio. В примере использован язык программирования C#, который позволяет производить высокоскоростные вычисления и обработку больших объемов данных. Для реализации алгоритмов обработки сигнала были использованы библиотеки с открытым исходным кодом. Полученные результаты демонстрируют возможности среды Visual Studio в обработке и анализе цифровых сигналов.

Библиографический список

1. Додобоев Н. Н., Кукарцева О. И., Тынченко Я. А. Современные языки программирования // Современные технологии: актуальные вопросы, достижения и инновации. 2014. №5. С. 81-85.
2. Магомадова З. С. Языки программирования высокого уровня // Разработка и применение наукоёмких технологий в эпоху глобальных трансформаций. 2020. №8. С. 94-96.
3. Садовников П. А., Дроздов А. Ю., Фонин Ю. Н. Библиотека параллельных функций цифровой обработки сигналов для интегрированного модуля на основе высокопроизводительного процессора цифровой обработки сигналов// Юный ученый. 2015. № 4 (67-1). С. 25-26.
4. Markovich I.I., Zavtur E.E. Digital processing methods of signals reflected from highly maneuverable aerial targets// Вестник воздушно-космической обороны. 2016. С. 17-23.