**«Санкт-Петербургский государственный электротехнический университет**

**«ЛЭТИ» им. В.И.Ульянова (Ленина)»**

**(СПбГЭТУ “ЛЭТИ”)**

|  |  |  |
| --- | --- | --- |
| **Направление** | 09.03.01 - Информатика и вычислительная техника | |
| **Программа** | Компьютерные технологии инжиниринга | |
| **Факультет** | Компьютерных технологий и информатики | |
| **Кафедра** | Систем автоматизированного проектирования | |
| *К защите допустить* |  | |
| Зав. кафедрой |  | Рыжов Н.Г. |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**МАГИСТРА**

**Тема: КОМПОНЕНТЫ САПР полиграфической системы аддитивной печати**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент |  |  |  | Мезенцев Р.С. |
|  |  | *подпись* |  |  |
| Руководитель | ассистент |  |  | Каримов А.И. |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |
| Консультанты | к.т.н., доцент |  |  | Иванов А.Н. |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |

Санкт-Петербург

2018

**ЗАДАНИЕ**

**на выпускную квалификационную работу**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой САПР |
|  | Рыжов Н.Г. |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | Мезенцев Р.С. | | | |  | Группа | 2301 | |
| Тема работы: | | | | | | | | |
| Место выполнения ВКР: кафедра САПР, СПбГЭТУ | | | | | | | | |
| Исходные данные (технические требования): | | | | | | | | |
| Перечень отчетных материалов: пояснительная записка, иллюстративный материал | | | | | | | | |
| Дополнительные разделы: | | | | | | | | |
|  | | | | | | | | |
| Дата выдачи задания | | | Дата представления ВКР к защите | | | | | |
| «01»\_ 2018 г. | | | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | | | |
|  | | |  | | | | | |
| Студент | |  | | Мезенцев Р.С. | | | |
| Руководитель | |  | | Каримов А.И. | | | |
|  | |  | |  | | | |

**календарный план выполнения**

**выпускной квалификационной работы**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой САПР |
|  | Рыжов Н.Г. |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | Мезенцев Р.С. |  | Группа | 2301 |
| Тема работы: КОМПОНЕНТЫ САПР ПОЛИГРАФИЧЕСКОЙ СИСТЕМЫ АДДИТИВНОЙ ПЕЧАТИ | | | | |

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование работ | Срок выполнения |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |

|  |  |  |
| --- | --- | --- |
| Студент |  | Мезенцев Р.С. |
| Руководитель |  | КаримовА.И. |

# РЕФЕРАТ

# АННОТАЦИЯ

# ABSTRACT

СОДЕРЖАНИЕ

[Определения, обозначения и сокращения 7](#_Toc511683936)

[Введение 7](#_Toc511683937)

[1. Обзор существующих технологий и решений 7](#_Toc511683938)

[1.1. Обзор решения 1 7](#_Toc511683939)

[1.2. Обзор решения 2 7](#_Toc511683940)

[1.3. Обзор решения 3 7](#_Toc511683941)

[Выводы 7](#_Toc511683942)

[2. Теоретическое описание алгоритмов 7](#_Toc511683943)

[2.4. Описание алгоритмов 1 7](#_Toc511683944)

[2.5. Описание алгоритмов 2 7](#_Toc511683945)

[2.6. Описание алгоритмов 3 7](#_Toc511683946)

[Выводы 7](#_Toc511683947)

[3. Реализация программного решения 7](#_Toc511683948)

[3.1. Выбор инструментов разработки 7](#_Toc511683949)

[3.2. Проектирование архитектуры 7](#_Toc511683950)

[3.3. Разработка 7](#_Toc511683951)

[3.4. Тестирование 7](#_Toc511683952)

[3.5. Отладка 7](#_Toc511683953)

[4. Составление бизнес-плана по коммерциализации результатов нир магистранта 7](#_Toc511683954)

[4.1. Концепция экономического обоснования разработки технического продукта 7](#_Toc511683955)

[4.2. Расчёт полных затрат при разработке программного модуля 7](#_Toc511683956)

[Заключение 7](#_Toc511683957)

[Список литературы 7](#_Toc511683958)

[ПРИЛОЖЕНИЕ А 8](#_Toc511683959)

# Определения, обозначения и сокращения

# Введение

# Обзор существующих технологий и решений

# Обзор решения eDavid

Робот eDavid создан командой разработчиков из Констацского университета (Германия). Изначально робот разрабатывался для сварки деталей автомобилей, но в конечном итоге нашел своё применение в живописи. Робот снабжен камерой, сенсорами и набором кистей (см. рисунок 1)

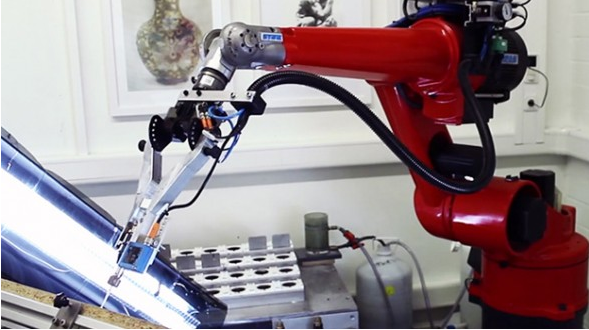


Рисунок 1 – Робот eDavid

Робот способен рисовать достаточно сложные композиционные картины, применяя при этом различные художественные направления. У робота 24-цветная палитра, робот работает только с акриловыми красками. Для создания картин робот использует набор из пяти кистей. Процесс рисования начинается с фотографирования оригинала. После этого робот анализирует фото, холст, размеры, выбирает подходящий цвет. На следующем этапе робот начинает наносить мазки, при этом применяется механизм обратной связи – имитация действия человека-художника. Каждые несколько минут робот делает фотографию своей работы и сравнивает её с оригиналом, после чего продолжает работу, либо корректирует последний шаг. Также во время оценки робот анализирует толщину, положение и цвет мазков, в результате чего выбирает кисть и соответствующий цвет из имеющейся палитры. Пример работы робота показан на рисунке 2.

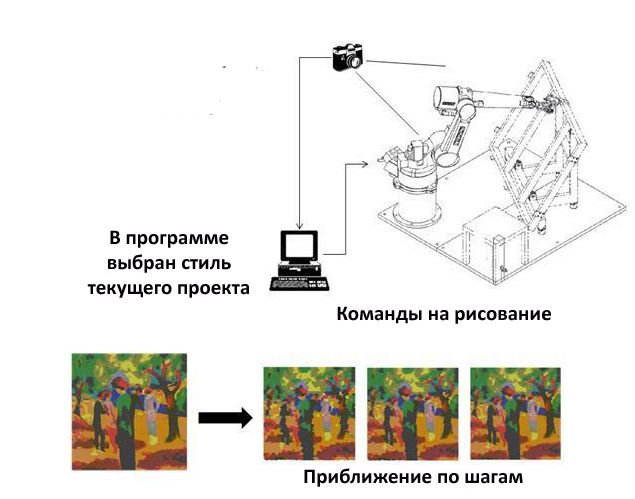


Рисунок 2 – Пример работы робота eDavid

Закончив работу, робот подписывает картину своим именем в нижнем правом углу. Примеры полученных результатов показаны на рисунке 3.



Рисунок 3 – Результаты работы робота eDavid

Преимуществами данного робота являются наличие механизма обратной связи, поддержка множества техник рисования, возможность работы с нескольким числом кисточек. Недостатками являются работа только с акриловыми красками, ограниченная палитра цветов, длительность рисования (до 10 часов).

# Обзор решения bitPaintr

Автором этого робота является голландский художник Пиндар Ван Армар. Изначально у робота должны были быть ассистенты, но в текущей пятой версии, построенной спустя 10 лет разработки, робот рисует картины самостоятельно (см. рисунок 4).



Рисунок 4 – Внешний вид робота bitPaintr

Ключевой особенностью этого проекта является применение специализированного искусственного интеллекта для создания портретов. Особенности используемых алгоритмов не раскрываются, но известно, что имеются настройки сложности и выбора стиля, которых всего 5 штук. Для начала рисования необходимо загрузить изображение и указать необходимые настройки. Процесс рисования занимает от 5 минут до 24 часов, при этом робот имеет возможность рисовать акриловыми и масляными красками (см рисунок 5).



Рисунок 5 – Результат работы робота bitPaintr

Максимальный размер картины составляет 30 на 42 дюйма. Проект изначально имел коммерческое направление применения, для чего автор разместил объявление на Kickstarter. В результате сбора средств была собрана сумма в 5665 долларов, целью было собрать 4800 долларов. Преимуществами этого робота является использование систем искусственного интеллекта, детализация полученных рисунков. Недостатками является время рисования картин, а также небольшое число стилей.

# Обзор решения AARON

Робот был создан в 1972 году Гарольдом Коэном – международно-известным художником, профессором Калифорнийского университета в Сан-Диего. Первые варианты робота рисовали черным и белым цветом, в последующих решениях применялся крупномасштабный струйный принтер на холсте.



Рисунок 6 – Внешний вид робота AARON

Разработка ПО изначально велась на C, но в начале 1990-ых годов ПО разрабатывалось на LISP. Стили рисования изображений закодированы, робот не имеет возможности самостоятельно изучать их. Программное обеспечение имеет циклы обратной связи, которые позволяют принимать решение относительно дальнейших действий. На самом верхнем уровне робот принимает решение, которое имеет отношение к композиции и организации изображения. На самом низком уровне он вычисляет инструкции для создания форм и линий, которые создают рисунок.



Рисунок 6 – Результат работы робота AARON

Автор робота утверждает, что творческая часть AARON является программным обеспечением независимо от того, какое устройство нанесения краски он присоединяет к нему, и при этом избегает называть AARON роботом.

Преимуществами робота является уникальный подход к рисованию – рисунки робота уникальны, никогда не повторяются за счет алгоритма с миллионами комбинаций. Недостатками являются отсутствие стилей для рисования, невозможность выбора исходного изображения для репродукции, использование принтера для рисования картины, а также сильная привязка к алгоритмам разработки.

# Выводы

Проведя анализ разработанных ранее решений, было установлено:

1. Практически все проекты скрывают свою внутреннюю логику работы программных компонентов;
2. Проекты сильно отличаются друг от друга как по возможностям, так и по характеристикам;
3. Ни один и рассмотренных проектов не способен работать полностью в автоматическом режиме.

# Теоретическое описание алгоритмов

# Описание алгоритмов фильтрации

Главной задачей алгоритмов фильтрации является улучшение качества изображения. В области обработки изображений чаще всего встречаются гауссов и импульсный шумы, а также их комбинации (см. рисунок 6).

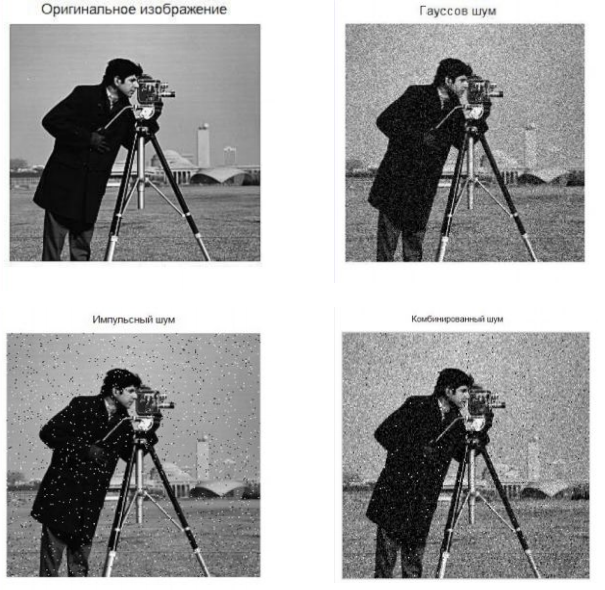


Рисунок 1.1 – Виды шума

На текущий момент не существует универсальных фильтров, которые подавляют все виды шумов. Самые распространенные методы удаления шумов:

1. Сглаживающие фильтры;
2. Медианные фильтры;
3. Ранжирующие фильтры;

Кроме этого, фильтры классифицируются на линейные и нелинейные. Для подавления гауссова шума могут использоваться оба типа. Линейный фильтр состоит из ядра фильтра, который может быть задан на растровом изображении. Фильтрация производится операцией дискретной свертки (взвешенного суммирования). К сожалению, применение линейных фильтров не дает возможность получить приемлемый вариант решения в широком спектре задач. Нужно принимать во внимание характер процессов кодирования, который как правило, является нелинейным, также, как и процессы передачи и восприятия информации, например, датчиков информации, зрительной системы человека и т.п.

Определение единой теории нелинейной фильтрации невозможно, в отличие от теории линейных методов фильтрации. Любой из перечисленных классов имеет свои преимущества и свою область применения. К примеру, известно, что лучшие результаты для сохранения перепадов оттенков, локальных пиков яркости, и других границ на изображениях может дать применение медианного фильтра.

Медианный фильтр использует нелинейный способ подавления шума. Основу медианного фильтра составляет апертура фильтра, которая состоит из нечетного числа отсчетов. Принцип работы заключается в следующем:

1. Определяется медианное значение элементов последовательности, т.е. находится средний по порядку член ряда, который образуется путем упорядочивания последовательности;
2. Значения, попадающие в маску, заменяются полученным медианным значением.

Медианный фильтр позволяет решать и задачи импульсного шумоподавления, и задачи подавления аддитивного шума, однако в случае подавления импульсного шума он является наиболее эффективным. Также его очень важной особенностью является сохранение перепадов яркости (см. рисунок 6).

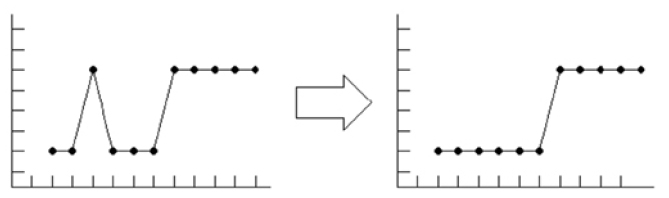


Рисунок 6 – Сохранение перепадов

Ранжирующий фильтр, также, как и сглаживающий, использует маску, причем маска может включать или не включать центральный пиксель. Элементы, которые попадают в маску, можно упорядочивать по возрастанию или по убыванию, и вычислить среднее значение дисперсии и интенсивности. Выходным значением фильтра является взвешенная сумма интенсивности центрального пикселя и медианы ряда.

# Описание алгоритмов вычисления векторов градиентов

Градиент изображения используется для реализации автоматического измерения координат точек снимков, выделение границ объектов на изображении, а также отделения фона изображения от самих объектов. Самыми известными и наиболее используемыми являются операторы Собеля, Робертса и Приютта. Эти алгоритмы базируются на преобразовании изображения с помощью скользящей маски – определенной группе пикселей, которые представляют собой квадратную матрицу (см. рисунок 6).

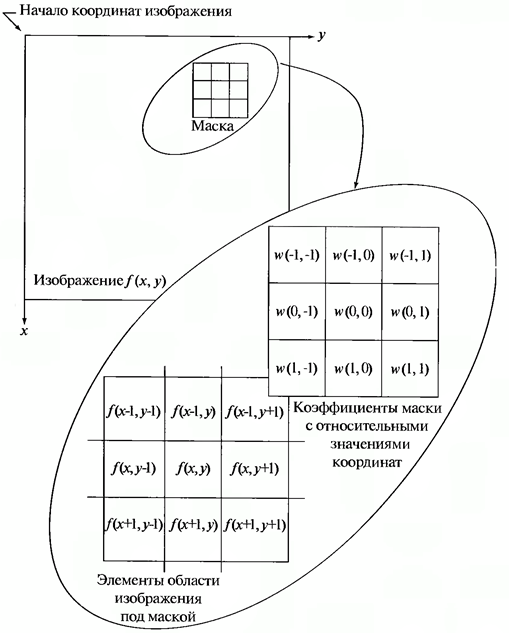


Рисунок 6 **–** Схема пространственной фильтрации

Алгоритм основан на обычном перемещении маски фильтра от одной точки к другой точке изображения; в каждой точке (x,y) отклик фильтра вычисляется с помощью предварительно заданных связей. Для линейной пространственной фильтрации отклик определяется суммой произведения коэффициентов фильтра на значения пикселей маски фильтра. Для маски 3х3 элемента, показанной на рисунке 6, отклик ***R*** линейной фильтрации в точке (x,y) изображения составит:

**image**(1.1)  
что, как видно, является суммой произведений коэффициентов маски на значения пикселей непосредственно под маской. Также можно заметить, что коэффициент w(0,0) находится при значении f(x,y), указывая тем самым, что маска центрирована в точке (x,y). При обнаружении перепадов яркости применяются дискретные аналоги производных первого и второго порядка.   
 Первая производная одномерной функции f(x) определяется как разность значений соседних элементов:

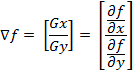
**image**(1.2)

Здесь попользуется запись в виде частной производной, благодаря чему сохраняются те же обозначения в случае с двумя переменными ***f(x,y)***, где необходимо иметь дело с частными производными по двум пространственным осям. Применение частной производной не изменяет существа рассмотрения.

Аналогично, вторая производная находится разностью соседних значений первой производной:

**image**(1.3)

Поиск первой производной изображения основан на различных дискретных приближениях двумерного градиента. По определению, градиент изображения f(x,y) в точке (x,y) — это вектор:

(1.4)

Направление вектора градиента совпадает с направлением максимальной скорости изменения функции f в точке (x,y) [2].

Также важную роль при обнаружении контуров играет модуль этого вектора, который обозначается ∇f и равен

image(1.5)

Эта величина является значением максимальной скорости изменения функции f в точке (x,y), где максимум достигается в направлении вектора ∇f. Величину ∇f также часто называют градиентом.

Направление вектора градиента является очень важной характеристикой. Обозначим α(x,y) угол между направлением вектора ∇f в точке (x,y) и осью x. Как известно из математического анализа,

image(1.6)

Отсюда можно легко найти направление контура в точке (x,y), которое перпендикулярно направлению вектора градиента в этой точке. Чтобы найти градиент изображения, достаточно вычислить величины частных производных ∂f/∂x и ∂f/∂y для каждой точки.

# Описание алгоритмов 3

# Выводы

# Реализация программного решения

# Выбор инструментов разработки

# Проектирование архитектуры

# Разработка

# Тестирование

# Отладка

# Составление бизнес-плана по коммерциализации результатов нир магистранта

# Концепция экономического обоснования разработки технического продукта

# Расчёт полных затрат при разработке программного модуля

# Заключение

# Список литературы

# ПРИЛОЖЕНИЕ А

Исходный код программного модуля