

РобоФинист 2024

Свободная творческая категория

Отчет по проекту «Автоматизация определения плотности вторичного сырья из пластика»
Команда «Регион52»
Старшая возрастная категория

Выполнили: Писаренко Егор, ученик 9 класса, МАОУ Лицей №38,

Каргапольцев Антон, ученик 9 класса, МАОУ Лицей №38

Научный руководитель:

Волкова Т. Н., педагог д/о МБУ ДО «ДДТ им. В. П. Чкалова»

Консультант: Руденко А. П., инженер ФНПЦ Нижегородского НИИ радиотехники

г. Н. Новгород,
2024

СОДЕРЖАНИЕ

Стр.

Аннотация.....	3
1.0 Введение	4
1.1 Технология переработки полимерных материалов	4
1.2 Цель.....	5
1.3 Задачи.....	5
2.0 Материалы и методики	6
2.1 Анализ существующих методов определения плотности пластика.....	6
2.2 Обоснование и выбор предлагаемого способа определения плотности пластика	8
2.3 Описание предлагаемого метода определения плотности пластика	9
2.4 Лабораторные исследования	9
2.5 Элементы функциональной схемы.....	11
2.6 Алгоритм.....	12
2.7 Структурная схема	13
2.8 Требования к программному обеспечению	15
2.9 Имитационная модель.....	15
2.10 Элементы предпринимательства	19
3.0 Результаты и их обсуждения	20
4.0 Заключение	21
Список литературы	22
Приложения.....	23

Аннотация

Цель:

В данной научной работе рассмотрена проблема сортировки мусора по плотности с целью выявления более простого и экономически выгодного способа.

Задачи:

1. Изучение и сравнительный анализ существующих методов определения плотности пластика.
2. Выбор и обоснование предлагаемого способа определения плотности пластика.
3. Рассмотрение проблем данного способа и путей их решения на практике.
4. Создание имитационной модели технологического процесса определения плотности.
5. Определение целевой аудитории для возможного применения предлагаемого способа.

Способ определения плотности пластика:

В этой работе плотность вычисляется путем нахождения массы и объема материала и их отношения. Преимущества этого способа заключаются в его простоте, надежности и высокой точности результата. Масса материала определяется путем взвешивания на весах. Следствием разрезания пластика на мелкие кусочки, которые следом поступают в измельчитель. С помощью компьютерного зрения выводятся линейные размеры измельченной насыпи, в следствие чего вычисляется объем.

Вывод:

Данный способ может быть применен в дальнейшей разработке методики вторичной обработки пластика, что позволит увеличить объем отечественного отсортированного вторичного сырья, а так же сократит материальные затраты на приобретение импортного сырья.

1.0 Введение

Достаточно примечательная история создания этого проекта. Изначально мы занимались робототехникой на среде ev3. Это нам очень нравилось и как то раз мы попробовали сделать проект связанный с сортировкой. Мы сортировали кубики по размеру и их цвету. Это был наш первый опыт, связанный с сортировкой. Поучаствовав во многих соревнованиях, в том числе заняв призовое место на международных соревнованиях в Санкт Петербурге, мы получили обучающий набор Arduino. С того момента мы углублялись в изучение Arduino, а так же языка программирования Python. Познакомившись с Python, мы узнали про библиотеку OpenCV, которую нам стало интересно изучить. Сначала мы пробовали получать изображение с различных камер, в том числе и телефона, потом мы начали обводить предметы, полученные с камер а после и вовсе мы начали подключать различные нейросети чтобы классифицировать объекты. Познакомившись с нейросетью, которая классифицирует мусор мы вспомнили про свой давний проект по сортировке. Мы загорелся идеей повторить подобное на основе Arduino, но сортировать уже твердые бытовые отходы с помощью компьютерного зрения.

С одной стороны нас заинтересовала идея вторичной переработки. Вторичное применение пластика является сейчас большой проблемой и пластик из мусора достаточно интересный ресурс для создания экологичных вещей. С другой стороны нас интересовал вопрос современных систем обработки изображения. Когда возникла проблема сортировки по плотности, у нас появилась идея попробовать применить современные методы цифровых технологий для сортировки по плотности.

1.1 Технология переработки полимерных материалов

Вторичная обработка полимерных материалов – это сложный многоуровневый процесс, который состоит из нескольких этапов.

Первым этапом в этом процессе является сбор мусора. От того, как именно

собирается и складывается мусор, напрямую зависит его пригодность к последующей переработке. Далее следует сортировка и разделение компонентов на категории. После сбора полимерных материалов, их необходимо правильно отсортировать. Для каждого вида пластика разработана собственная технология переработки. Наиболее важным параметром для выбора технологии переработки и применения является плотность пластика. Различные типы пластика имеют разные плотности, поэтому этот этап позволяет эффективно разделить их на группы. На следующем этапе, из полимерных отходов удаляются все загрязнения, которые могут повлиять на результаты вторичной переработки. Пластик очищается от клея, пищи, этикеток и других компонентов. Далее, предварительно подготовленный мусор, необходимо пропустить через специальные дробилки. Они преобразуют отходы в единую массу из мелких фракции. Отсортированные материалы могут быть направлены на переработку по отдельности, что увеличивает эффективность процесса переработки и помогает сократить отходы. Разделение пластиковых материалов по плотности или определение их плотности - это важный процесс в переработке пластика.

1.2 Цель

В данной научной работе рассмотрена проблема сортировки мусора по плотности с целью выявления более простого и экономически выгодного способа.

1.3 Задачи

В качестве задач для данной научной работы были выбраны следующие пункты:

1. Изучение и сравнительный анализ существующих методов определения плотности пластика.
2. Выбор и обоснование предлагаемого способа определения плотности пластика.
3. Рассмотрение проблем данного способа и путей их решения на практике.
4. Создание имитационной модели технологического процесса определения плотности.
5. Определение целевой аудитории для возможного применения предлагаемого способа.

2.0 Материалы и методики

В этой части исследовательской работы рассматриваются существующие методы определения плотности пластика, их преимущества и недостатки. Далее предлагается новый, усовершенствованный способ определения плотности. Данный способ требует измельчения пластика до мелких частиц для наиболее точного определения плотности с помощью технологии компьютерного зрения.

2.1 Анализ существующих методов определения плотности пластика

Есть несколько сформировавшихся методов определения плотности пластика, и вот некоторые из них:

1. Центрифугирование: В этом методе материалы подвергаются вращению в центрифуге. Благодаря силе инерции более плотные материалы оказываются ближе к центру, тогда как менее плотные материалы остаются ближе к краю и могут быть отделены.

- **Преимущества:**

Быстрота: Данный метод позволяет провести измерения относительно быстро.

Применимость к различным материалам: Метод подходит для различных типов материалов.

- **Недостатки:**

Требует специализированного оборудования: Центрифуга должна быть соответствующей мощности и все точности.

Ограничения по размеру образца: Возможны ограничения по размеру образца для проведения измерения.

2. Флотационные установки: Этот метод основан на различиях во взаимодействии разных материалов с воздушными пузырьками в жидкости.

Плотные материалы будут быстрее погружаться, в то время как менее плотные будут задерживаться на поверхности.

- Преимущества:

Высокая точность: Флотационные установки могут обеспечить точное измерение плотности материалов.

Применимость для разнообразных веществ: Подходит для различных типов материалов, включая пластик.

- Недостатки:

Сложность настройки: Требуется определенный уровень экспертизы для правильной настройки и интерпретации результатов.

Требует специализированного оборудования: Для проведения флотации нужны специальные установки.

3. Гидростатическое взвешивание: Этот метод используется для определения плотности пластиковых материалов. При этом образец погружается в жидкость, и по объему вытесненной жидкости можно определить плотность материала.

- Преимущества:

Простота: Относительно простой метод измерения плотности.

Не требует сложного оборудования: Для измерения плотности пластика в воде можно использовать стандартные гидростаты.

- Недостатки:

Возможны ошибки: Точность результатов может зависеть от правильного выполнения измерений.

Влияние воздушных пузырьков: Наличие воздушных пузырьков на поверхности образца может исказить результаты.

4. Пикнометрия: В этом методе используют стеклянные сосуды (пикнометры), заполненные до краев жидкостью. Путем взвешивания с пикнометром и без него можно определить плотность материала.

- Преимущества:

Высокая точность: Пикнометрия обеспечивает высокую точность измерения

плотности материалов.

Расширенная применимость: Метод подходит для различных материалов и форм.

- Недостатки:

Сложность метода: Требуется определенного уровня экспертизы и аккуратности при проведении измерений.

Времязатратность: Измерения могут быть более времязатратными по сравнению с некоторыми другими методами.

2.2 Обоснование и выбор предлагаемого способа определения плотности пластика

Практически каждый из рассмотренных способов отличается дороговизной оборудования и сложностью регулировок и настроек.

Для целей данной работы решено было воспользоваться традиционным методом. Этот метод основан на измерении массы и объема материала непосредственно. Плотность вычисляется как отношение массы к объему.

Этот способ обладает следующими преимуществами:

- Простота
- Надежность
- Наиболее точный способ: При правильных измерениях этот метод может

обеспечить высокую точность результатов.

2.3 Описание предлагаемого метода определения плотности пластика

Для определения плотности необходимо найти массу и объем. Массу можно определить путем взвешивания на весах. А вот определение объема представляет из себя достаточно сложный процесс:

Пластик разрезается на мелкие кусочки, которые далее плотно собираются вместе. Собранные кусочки пластика поступают на транспортер.

Измерение объема вычисляется посредством получения линейных размеров измельченной насыпи в процессе обработки изображения с помощью компьютерного зрения.

2.4 Предварительные испытания

Для определения возможности измерений объема объекта производились несколько испытательных действий:

1. Делался снимок выбранного тест-объекта.
2. Изображение передавалось на компьютер.
3. Изображение обрабатывалось в программе, написанной на языке Python, использовалась библиотека «OpenCV». Применяя предварительно обученную имеющуюся нейросеть «YOLO» выделялся необходимый объект контурами на экране компьютера в режиме реального времени.
4. По контурам объекта определялась его площадь в пикселях.
5. Измерялась так же реальная площадь объекта. После сопоставления реальной площади и площади в пикселях находился коэффициент для расстояния от камеры до объекта, с которого взято изображение.
6. Полученная площадь A в миллиметрах квадратных умножаем на фиксированную ширину B в миллиметрах, получив объем (См. Рис. 1).

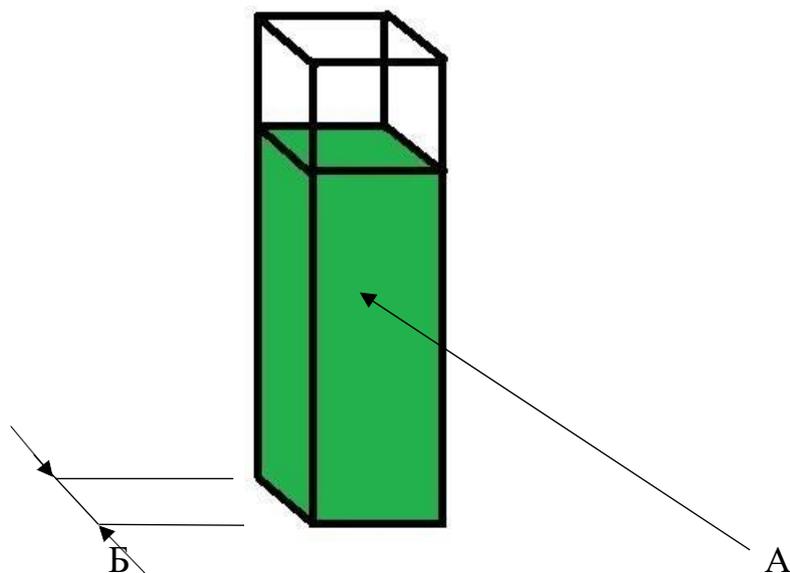


Рис. 1

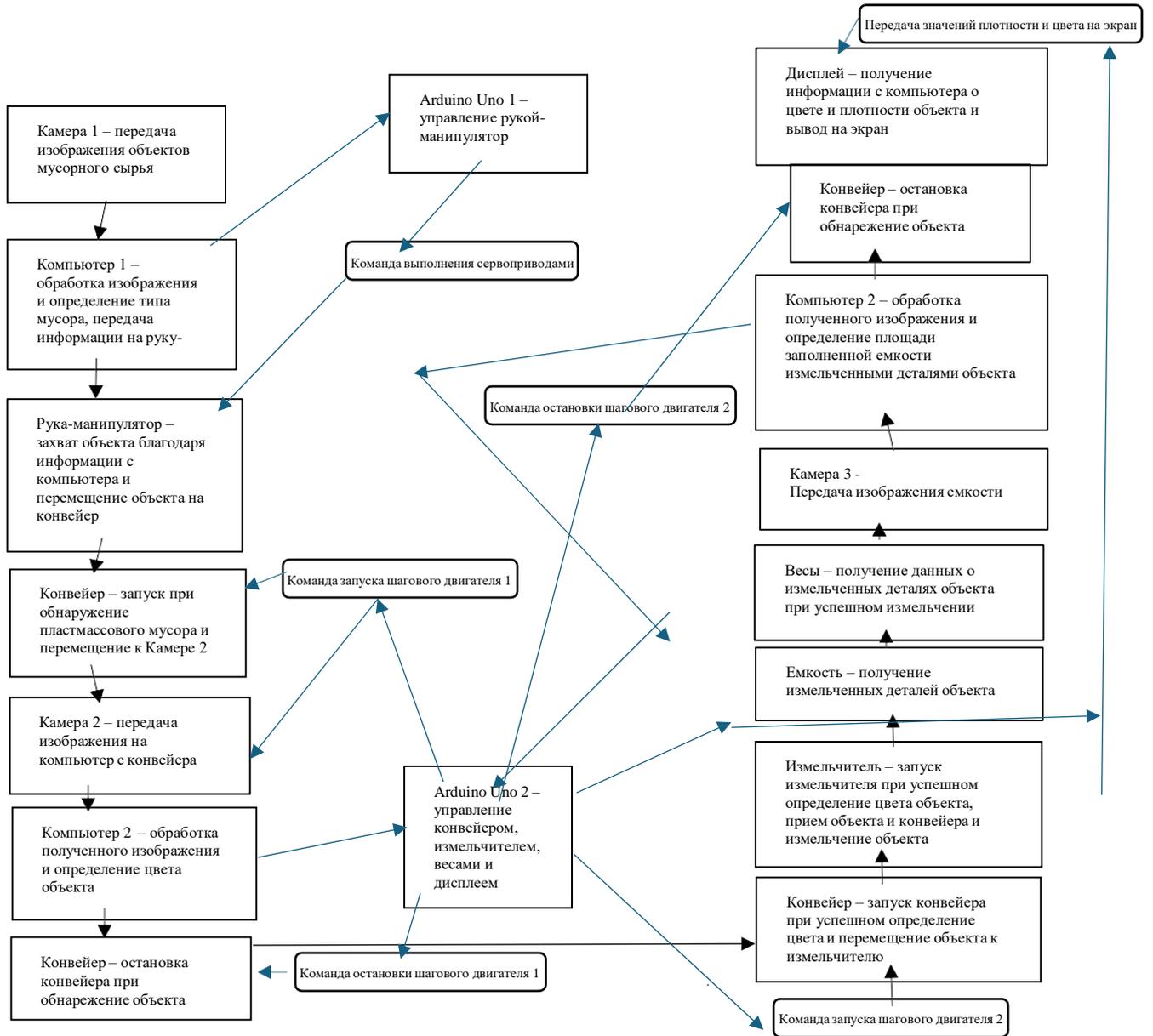
Вычисленный объем позволяет вычислить плотность, которая может быть зафиксирована в необходимом устройстве.

Для отработки методики, представленной в данном проекте была создана имитационная модель части технологического процесса сортировки пластика для вторичной переработки.

2.5 Элементы функциональной схемы

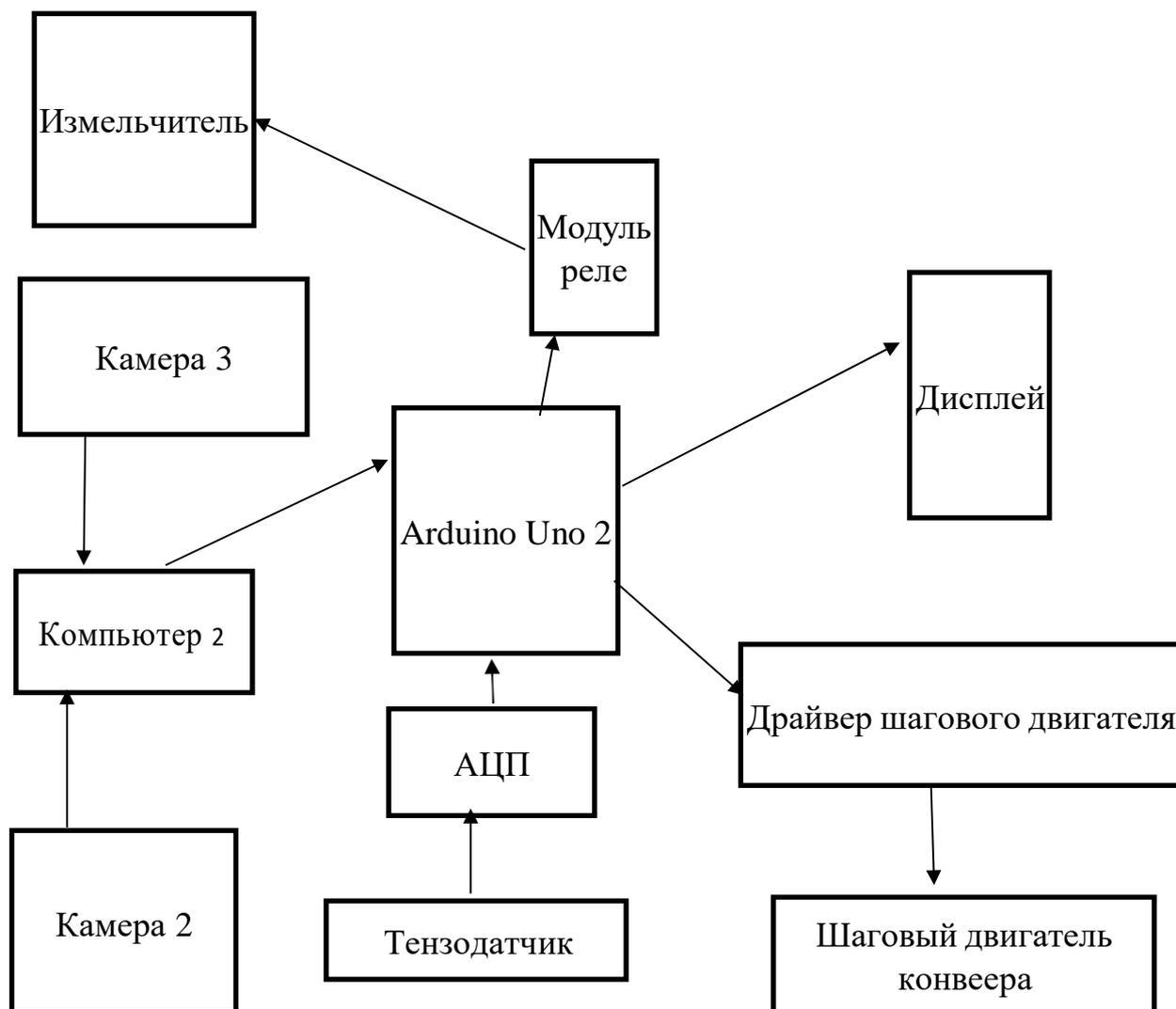
Функциональные Элементы	Действия
Камера 1	Сканирует множество мусорных объектов и передает изображение на компьютер 1
Компьютер 1	Обрабатывает изображение, полученное с камеры 1, определяет тип мусора с изображения, передает информацию на Arduino Uno
Рука-манипулятор	Получает команды с Arduino Uno, захватывает объект
Arduino Uno 1	Выполняет передачу команд с компьютера; получает информацию с компьютера
Конвейер	Отвечает за перемещение объекта, получает команды с Arduino Uno 2
Камера 2	Сканирует объект на конвейере и передает изображение на компьютер 2
Компьютер 2	Обрабатывает изображение, полученное с камеры 2, определяет цвет объекта на конвейера с изображения; обрабатывает изображение, полученное с камеры 3, определяет площадь заполненного объема емкости изображения, полученного с камеры 3; передает информацию на Arduino Uno
Камера 3	Сканирует состояние емкости и передает изображение на компьютер 2
Измельчитель	Измельчает объект, полученный с конвейера; получает команды с Arduino Uno
Весы	Измеряет массу измельченных объектов в емкости; передает информацию на Arduino Uno
Arduino Uno 2	Выполняет передачу команд с компьютера; получает информацию с компьютера

2.6 Алгоритм

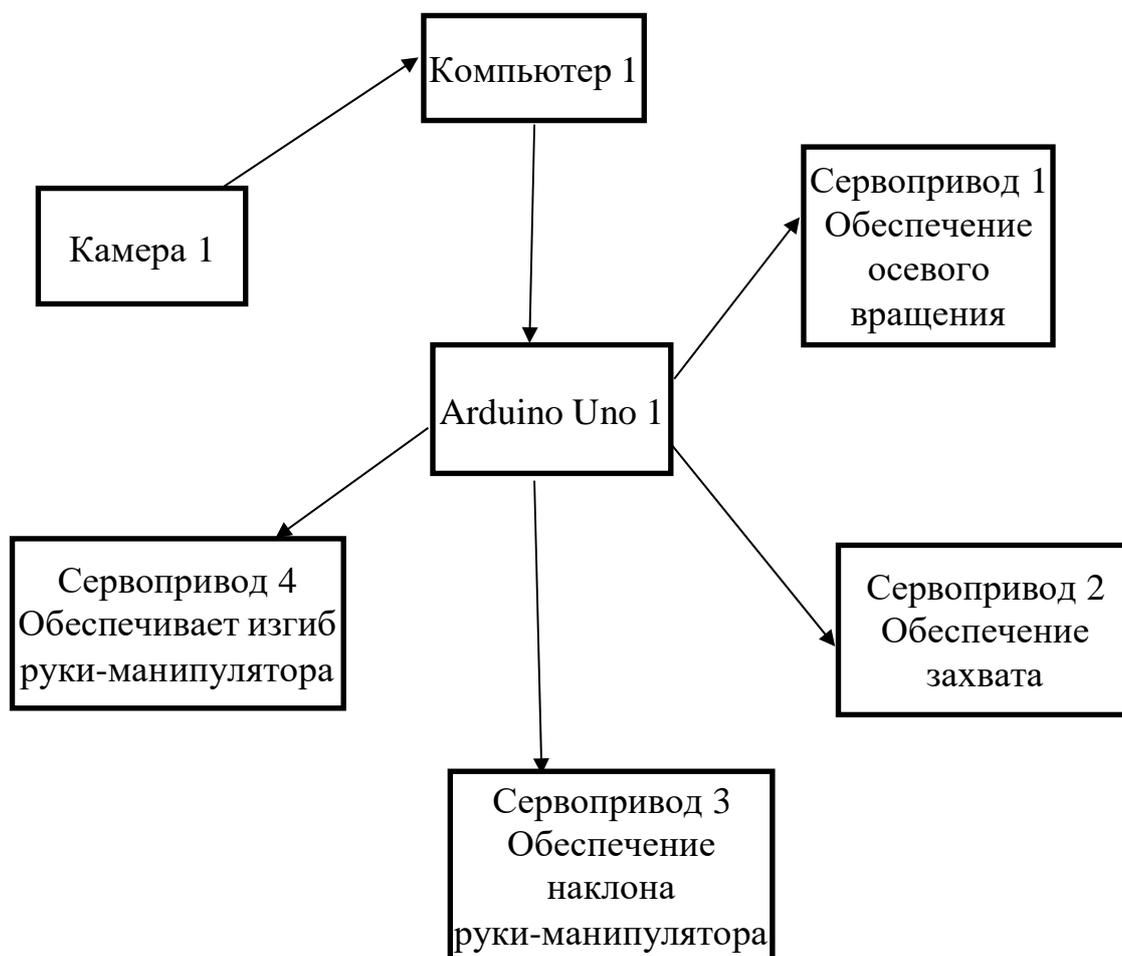


2.7 Структурная схема

Задача 1: определение плотности



Задача 2: отбор пластика из мусора



2.8 Требования к программному обеспечению

Для реализации проекта потребуется: компьютер со средой Python, установленная библиотека OpenCV для работы с изображениями; YOLO для обнаружения объектов, позволяет быстро и точно определять объекты на изображениях или видео; pySerial для COM-порта необходим для обмена данными через COM-порт, что позволит взаимодействовать с внешними устройствами, такими как Arduino; numpy для математики, pandas для данных, Arduino IDE использовать для программирования платы Ардуино, AccelStepper для шаговых двигателей, HX711 для весов.

2.9 Имитационная модель

Модель состоит из нескольких частей. За выбор пластмассового мусора отвечает роботизированная рука-манипулятор. Техническое описание нашей роботизированной руки включает в себя использование микроконтроллера ArduinoNano в качестве основы для управления устройством. Рука состоит из нескольких сегментов, соединенных с помощью винтов с гайками. Четыре сервопривода, обеспечивающих плавное и точное движение. Детали руки изготовлены с использованием 3D-печати, что позволяет создавать сложные геометрические формы и обеспечивает легкость и прочность конструкции. Для расширения функциональности руки применяется специальная плата расширения, которая позволяет подключать дополнительные моторы. Также в конструкцию добавлена камера для визуального восприятия окружающей среды. Эта интеграция технологий обеспечивает роботизированной руке возможность выполнять разнообразные задачи, такие как захват и перемещение объектов, выполнение прецизионных операций и взаимодействие с окружающей средой с высокой степенью точности и гибкости. Все эти элементы совместно создают мощное и эффективное устройство, способное решать различные задачи в области робототехники и автоматизации.

Обнаруженный пластиковый мусор перемещается на транспортёр (конвейер). Корпус конвейера состоит из железных уголков, которые жестко закреплены шурупами к толстой деревянной доске. Этот дизайн обеспечивает прочную и устойчивую основу для работы конвейера даже при высокой нагрузке. Между уголками расположены трубки, которые поддерживают и направляют движение конвейерной ленты. Дополнительную жесткость всей конструкции обеспечивают длинные железные стержни, закрепленные между уголками. Шаговой двигатель Nema17 отвечает за движение конвейерной ленты. Он точно и плавно перемещает конвейерную ленту, обеспечивая необходимую скорость и точность работы всей системы. Для преобразования движения от шагового двигателя к конвейерной ленте

используется замкнутый зубчатый ремень GT2 6мм. Он передает вращательное движение от зубчатого шкива GT2-6, 16 зубьев, прикрепленного к шаговому двигателю, на зубчатый шкив GT2, 80 зубьев, присоединенный к железному стержню, который поддерживает трубки и конвейерную ленту. Микроконтроллер Arduino Uno в связке с CNC SHIELD V3 играют ключевую роль в управлении шаговым двигателем. Они обеспечивают точное управление скоростью и направлением движения конвейера, что необходимо для эффективной работы всей системы. Конвейерная лента выполнена из ленточного экспандера из высокопрочного латекса, ширина которого 15 см.

С транспортёра мусор поступает в измельчитель, который играет ключевую роль в переработке материалов. Измельчитель – специально созданное оборудование. Он конструктивно создан из молярной кюветы диаметром 15 см, также в качестве измельчающего элемента установлено лезвие от старой «болгарки» (угловой шлифовальной машины для резки). Все элементы измельчителя надежно закреплены железными болтами, обеспечивая прочность и безопасность работы оборудования. Для транспортировки измельченного пластика в измерительную емкость применяется сворная металлическая направляющая. Следом материал направляется в измерительную емкость – тонкая прозрачная пластиковая емкость, где он собирается для дальнейшей обработки или использования. Измельченная масса взвешивается на весах. Весы представляют собой самодельно собранную модель.

Элементы управления:

Видео-камера 1, находящаяся на захвате манипулятора, обработка поступающего сигнала происходит на компьютере. Esp32-cam – компактный модуль, основанный на чипе esp32 и предназначенный для создания устройств с подключением к Wi-Fi. Он представляет собой удобное решение для создания систем видеонаблюдения. Для управления транспортером используется плата Arduino Uno с расширительной платой CNC Shield V3, благодаря которой управляется шаговый мотор Nemo 17, а также приводится в действие зубчатый ремень. Измельчитель запускается с помощью одноконтактной реле, подключенной к плате Arduino Uno.

После полной остановки измельчителя, измельченная масса поступает в измерительную емкость, рядом с которой установлена камера представленная esp32-sam, благодаря которой компьютер получает и обрабатывает изображение используя OpenCV и нейронные сети, получая информацию о пространстве занятом в емкости измельченными частицами, после чего площадь домножается на глубину емкости и вычисляется объем. Информация о цвете и объеме передается на плату Arduino Uno. Для определения массы измельченных объектов, уже находящихся в прозрачной емкости, используются весы, играющие важную роль в процессе переработки материалов. Весы состоят из тензодатчика, который закреплен между двумя пластиковыми платформами. Этот компонент помогает точно измерить массу материала. Для связи между Arduino Uno и тензодатчиком ответственен драйвер HX711, который обеспечивает удобный и надежный способ передачи данных о массе объектов.

Стоимость частей имитационной модели:

• Вспомогательные детали:

1. Фанера (3 листа) - 900 рублей
2. Металлический крепежный уголок (4 штуки) - 30 рублей
3. Пластина соединительная крепежная (10 штук) - 520 рублей
4. Крепежные изделия (64 штуки) - около 80 рублей (в зависимости от их вида)

Стоимость всех компонентов подгруппы: 1530 рублей

• Конвейер:

1. Зубчатый шкив GT2-6, 16 зубьев, 5мм - 250 рублей
2. Зубчатый шкив GT2, 80 зубьев, посадка 10мм, для ремня 6мм - 250 рублей
3. Плата Arduino Uno - 400 рублей
4. Шаговый двигатель Nema17 - 300 рублей
5. Лента латексная - 200 рублей
6. Пластиковые трубки - 200 рублей

Стоимость всех компонентов подгруппы: 1600 рублей

• Измельчитель:

1. Емкость для обработки материала - 300 рублей
2. Емкость для сбора обработанного материала - 100 рублей
3. Модуль реле 5В, 10А, 1 канал - 120 рублей
4. Угловая шлифовальная машина (болгарка) – 1500 рублей

Стоимость всех компонентов подгруппы: 2020 рублей

- Механическая рука

1. Части руки, напечатанные на 3д принтере (14 штук) - 2000 рублей
2. Сервоприводы Arduino SG90 (3 штуки) - 500 рублей
3. Камера esp32-cam - 500 рублей
- 4 Плата Arduino Nano - 300 рублей
5. Расширительная плата Sensor Shield под Arduino Nano - 400 рублей
6. Сервоприводы MG996R Servo (3 штуки) - 1450 рублей

Стоимость всех компонентов подгруппы: 5150 рублей

- Механические весы:

1. Ацп НХ711 - 180 рублей
2. Тензометрический датчик на 1кг - 250 рублей

Стоимость всех компонентов подгруппы: 430 рублей

- Средство для вывода информации:

1. Дисплей LCD2004 для Arduino - 300 рублей

- Средство определения объема:

1. Камера esp32-cam - 500 рублей

Общая стоимость: 11510 рублей

2.10 Элементы предпринимательства

Swot анализ:

<p>Сильные стороны:</p> <ol style="list-style-type: none">1. Простота в разработке2. Низкая цена на все нужные материалы.3. Достаточно точный результат на выходе.4. Компактность модели.5. Полезность проекта для окружающей среды.	<p>Слабые стороны:</p> <ol style="list-style-type: none">1. Слаборазвитые пункты сбора.2. Малый ассортимент предлагаемых видов переработки.3. Недостаток собственных финансовых ресурсов.
<p>Возможности:</p> <ol style="list-style-type: none">1. Выход продукта на международный рынок.2. Преобразование отдельных частей продукта.3. Использование неполного технологического цикла.4. Получение государственной поддержки.	<p>Угрозы:</p> <ol style="list-style-type: none">1. Увеличение конкуренции.2. Отсутствие согласованной службы сбора.3. Срыв поставок оборудования.

3.0 Результаты и их обсуждение

В работе продемонстрирована модель оборудования, которая упростит и сделает выгодней процесс переработки вторичного сырья.

На данный момент выполнена лишь часть всей модели: конвейер и программа, которая управляет конвейером; рука-манипулятор и программа, управляющая рукой-манипулятором; программа, которая координирует работу камер для определения типа мусора, цвета объекта и площади объекта. Эта работа заняла примерно месяц.

В будущем предстоит сконструировать измельчитель, емкость для сбора измельченных частей объекта, прописать программу для настройки работы измельчителя; связать конструктивно и с помощью программы все устройства между собой: камеру для определения типа мусора, руку-манипулятор, конвейер, камеру для определения цвета объекта, измельчитель, емкость для сбора измельченных частей объекта, камеры для определения площади объекта и весы.

4.0 Заключение

В настоящее время российские заводы для переработки вторичного сырья закупают его за рубежом.

Проведя серию испытаний следует вывод, что данный способ может быть применен в дальнейшей разработке методики вторичной обработки пластика, что позволит увеличить объем отечественного отсортированного вторичного сырья, а также сократит материальные затраты на приобретение импортного сырья.

Список литературы

Электронные ресурсы:

Al-Salem, S. M., Lettieri, P., & Baeyens, J. (2009). Recycling and recovery routes of plastic solid waste (PSW): A review. *Waste Management*, 29(10), 2625-2643. DOI: [10.1016/j.wasman.2009.06.004](https://doi.org/10.1016/j.wasman.2009.06.004)

De Gruyter (2020). Recycling of Plastics: A Review. DOI: [10.1515/revce-2021-0026](https://doi.org/10.1515/revce-2021-0026)

European Commission. (2020). Improving the economics of plastic recycling. Source <https://musor.moscow/blog/pererabotka-plastika/>

Industrial Plastic's Economics Research Council. (2018). Plastics Industry in the 21st Century: The Economic and Environmental Impact. [Report] Plastic News

Recycle Nation. (n.d.). Why Density Separation is Important in Plastic Recycling. Source

Zabaniotou, A., & Stavropoulos, G. (2012). Recycling of polymer materials. Source Proper plastic recycling: clarification of labeling codes. Source

Приложения

Qr code на репозиторий Github:



Коды программ:

Код работы имитационной модели с получением данных от Python.

```
// Включает библиотеку LiquidCrystal_I2C.
#include <LiquidCrystal_I2C.h>
// Включает библиотеку Wire.
#include <Wire.h>

// Определяет пины для управления шаговыми двигателями, реле и весами.
#define EN 8
#define X_DIR 5
#define X_STP 2
#define RELAY_IN 12

// Включает библиотеку HX711.
#include "HX711.h"

// Устанавливает пины для датчика веса HX711.
// HX711.DOUT - pin #A1
// HX711.PD_SCK - pin #A0
const int HX711_DOUT_PIN = A10;
const int HX711_PD_SCK_PIN = A11;

// Создает объект scale класса HX711.
HX711 scale(HX711_DOUT_PIN, HX711_PD_SCK_PIN);

// Устанавливает коэффициент наклона для датчика веса.
// Это значение получено путем калибровки весов с известными грузами; см. README для получения
// подробностей
scale.set_scale(-465.9);

// Сбрасывает показания датчика веса.
scale.tare();

// Инициализирует ЖК-дисплей lcd.
LiquidCrystal_I2C lcd(0x27,16,2);

// Включает подсветку ЖК-дисплея.
lcd.init();
lcd.backlight();

// Начинает последовательную передачу данных со скоростью 9600 бод.
```

```

scale.tare();

// Инициализирует ЖК-дисплей lcd.
lcd.init();
lcd.backlight();

// Начинает последовательную передачу данных со скоростью 9600 бод.
Serial.begin(9600);

// Устанавливает пин RELAY_IN как выход.
pinMode(RELAY_IN, OUTPUT);

// Устанавливает пин X_DIR как выход.
pinMode(X_DIR, OUTPUT);

// Устанавливает пин X_STP как выход.
pinMode(X_STP, OUTPUT);

// Устанавливает пин EN как выход и выключает его.
pinMode(EN, OUTPUT);
digitalWrite(EN, LOW);

// Устанавливает начальное время для отслеживания интервалов.
startTime = millis();
}

// Основной цикл программы.
void loop() {

// Пауза перед запуском.
delay(20);

// Проверяет, прошло ли указанное количество времени с момента последнего вызова функции.
if (millis() - time > interval) {

// Получение сигнала с

// Отправляет сообщение по последовательному порту.
Serial.println("00000");
}
}

```

```

Serial.begin(9600);

// Устанавливает пин RELAY_IN как выход.
pinMode(RELAY_IN, OUTPUT);

// Устанавливает пин X_DIR как выход.
pinMode(X_DIR, OUTPUT);

// Устанавливает пин X_STP как выход.
pinMode(X_STP, OUTPUT);

// Устанавливает пин EN как выход и выключает его.
pinMode(EN, OUTPUT);
digitalWrite(EN, LOW);

// Устанавливает начальное время для отслеживания интервалов.
unsigned long startTime = millis();

// Перемещает шаговый двигатель в определенном направлении указанное количество шагов.
void step(boolean dir, byte dirPin, byte stepperPin, int steps) {
digitalWrite(dirPin, dir);
pinMode(RELAY_IN, OUTPUT);
for (int i = 0; i < steps; i++) {
digitalWrite(stepperPin, HIGH);
delayMicroseconds(stepDelay);
digitalWrite(stepperPin, LOW);
delayMicroseconds(stepDelay);
}
}

// Настраивает параметры шагового двигателя.
void setup() {

// Начинает последовательную передачу данных со скоростью 38400 бод.
Serial.begin(38400);

// Устанавливает коэффициент наклона для датчика веса.
scale.set_scale(-465.9);

// Сбрасывает показания датчика веса.
}
}

```

```

// Перемещает шаговый двигатель влево (положительное направление) указанное количество шагов.
while (qwe < 8) {
    Serial.println("11111");
    qwe += 1;
    step(true, X_DIR, X_STP, steps);
}

// Устанавливает реле в состояние включения.
Serial.println("222222");
qwe = 1;

// Включает реле.
delay(15000);

// Отключает реле.
digitalWrite(RELAY_IN, HIGH);

// Перемещает шаговый двигатель вправо (отрицательное направление) указанное количество шагов.
while (qwe < 6) {
    qwe += 1;
    digitalWrite(RELAY_IN, LOW);
    delay(5000);
    delay(8000);
}

// Сбрасывает значение переменной.
qwe = 1;

// Устанавливает время для отслеживания интервала.
time = millis();

// Пауза перед выводом на экран.
delay(1000);

// Ожидание получения данных от Python.
while (receivedFloat < 1) {
    if (Serial.available() > 0) {
        float data;
        Serial.readBytes((char *)&data, sizeof(data));
        receivedFloat = data;
    }
}

```

```

}
}

// Рассчитывает плотность.
massa = scale.get_units(10);
density = massa / receivedFloat;
// Выводит значение плотности на ЖК-дисплей.
while (qwe < 10) {
    lcd.setCursor(0, 0);
    lcd.print("color:   white");
    lcd.setCursor(0, 1);
    lcd.print("Density:");
    lcd.setCursor(8, 1);
    lcd.print(density);
    qwe += 1;
}

// Сбрасывает значение переменной.
qwe = 1;

// Пауза перед вторым запуском.
delay(1000000);
}
}

```

Код работы манипулятора с получением данных от Python.

```
1 #include <servo.h>
2
3 Servo dno;
4 Servo plecho;
5 Servo lokot;
6 Servo kist1;
7 Servo kist2;
8 Servo ruka;
9
10 float receivedFloat = 0; // Переменная для хранения данных, полученных от Python
11
12 void setup() {
13 // attach each servo object to a pin number
14 // Замените 100, 1, 10, 8, 3, 4, 6 на ваши пины для сервоприводов!
15 dno.attach(13);
16 plecho.attach(10);
17 lokot.attach(8);
18 kist1.attach(3);
19 kist2.attach(4);
20 ruka.attach(6);
21 pinMode(13, OUTPUT);
22
23 // Начальное положение сервоприводов
24 dno.write(0);
25 delay(1000);
26 plecho.write(100);
27 delay(1000);
28 lokot.write(90);
29 delay(1000);
30 kist1.write(90);
31 kist2.write(90);
32 delay(1000);
33
34 // Включение светодиода на пине 13
35 digitalWrite(13, HIGH);
36
37 // Плавные движения сервоприводов
38 sweep(kist2, 90, 130, 20);
39 sweep(plecho, 100, 160, 20);
40 sweep(lokot, 90, 180, 20);
41 delay(2000);
42
43 // Ожидание первой команды от Python
44 while (receivedFloat < 1) {
45 if (Serial.available() > 0) {
46 float data;
47 Serial.readBytes((char *)&data, sizeof(data));
48 receivedFloat = data;
49 }
50 }
51 }
```

1

```
54 // Проверка команды
55 if (receivedFloat == 10) {
56 // Последовательность поворотов сервоприводов
57 beru();
58 delay(2000);
59 } else {
60 receivedFloat = 0;
61 // Другая последовательность поворотов сервоприводов
62 sweep(dno, 0, 60, 20);
63 // Ожидание команды от Python
64 while (receivedFloat < 1) {
65
66 if (Serial.available() > 0) {
67 float data;
68 Serial.readBytes((char *)&data, sizeof(data));
69 receivedFloat = data;
70 }
71 }
72
73 // Проверка команды
74 if (receivedFloat == 10) {
75 // Последовательность поворотов сервоприводов
76 beru();
77 delay(2000);
78 }
79
80 // Включение светодиода
81 digitalWrite(13, HIGH);
82
83 }
84
85 void loop() {
86 // Ничего не происходит в loop()
87 }
88
89 void sweep(Servo servo, int oldPos, int newPos, int servospeed) {
90 if (oldPos <= newPos){
91 for (oldPos; oldPos <= newPos; oldPos += 1){
92 servo.write(oldPos);
93 delay(servospeed);
94 }
95 } else if (oldPos >= newPos){
96 for (oldPos; oldPos >= newPos; oldPos -= 1){
97 servo.write(oldPos);
98 delay(servospeed);
99 }
100 }
101 }
102
103 void beru() {
104 sweep(plecho, 160, 85, 20);
105 sweep(kist2, 130, 130, 20);
106 sweep(ruka, 180, 120, 20);
107 sweep(plecho, 85, 110, 20);
108 sweep(kist2, 130, 160, 20);
109 sweep(dno, 0, 180, 20);
110 sweep(kist2, 160, 130, 20);
111 sweep(ruka, 120, 180, 40);
112 sweep(kist2, 130, 160, 20);
113 sweep(lokot, 180, 150, 20);
114 }
```

2

Определение объема и отправка данных на Arduino:

```

import numpy as np
import cv2
import serial
import struct
import time

# Read float values from file as to labels
url = "http://172.20.10.6:4242/video"
cap = cv2.VideoCapture(url)

# Set video resolution (optional)
cap.set(3, 300)
cap.set(4, 300)

# Initialize variables
area = 0
ms = 0

# Initialize serial connection
ser = serial.Serial('/dev/ttyUSB0', 9600) # Replace with your serial port
time.sleep(2) # Wait for serial port to be ready

while True:
    # Read frame from frame
    ret, frame = cap.read()

    if not ret:
        print("Error: failed to capture frame.")
        break

    # Convert the frame to HSV color space
    hsv = cv.cvtColor(frame, cv.COLOR_BGR2HSV)

    # Define the lower and upper bounds for the color
    lower_hsv = np.array([0, 0, 200])
    upper_hsv = np.array([180, 255, 255])

    # Threshold the HSV image to get only blue colors
    mask = cv.inRange(hsv, lower_hsv, upper_hsv)

    # Find contours in the thresholded image
    cnts = cv.findContours(mask.copy(), cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE)
    msk = cv.cvtColor(mask, cv.COLOR_GRAY2_BGRA)

    # Loop over the contours
    for c in cnts:
        # Calculate area of the contour
        area = cv.contourArea(c)

        # Draw small contour
        if area > 1000: # adjust this threshold as needed
            # Draw center marker
            cv.drawMarker(frame, c[0], (0, 255, 0), 5)

            # Calculate centroid of the contour
            (cx, cy, mc) = cv.moments(c)
            cx = int(cx/area)
            cy = int(cy/area)

    # Draw centroid
    cv.drawMarker(frame, (cx, cy), (0, 255, 0), 5)
    cv.putText(frame, "Center", (cx - 20, cy - 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

    # Display the resulting frame
    cv.imshow(" Webcam", frame)

    # Update maximum area
    if area > ms:
        ms = area
        print("Area:", ms)

    # Send data to Arduino
    float_var = ms
    ser.write(struct.pack('f', float_var))
    time.sleep(1) # Wait for Arduino to process data

    # Wait for 1 second
    time.sleep(1)

    # Increment loop counter
    ms += 1

    # Check for key press; if 'q' is pressed, exit the loop
    if cv.waitKey(1) & amp; not == ord('q'):
        break

# Close serial port connection
ser.close()

# Release the video capture object and close all windows
cap.release()
cv.destroyAllWindows()

```

1

```

# Draw centroid
cv.drawMarker(frame, (cx, cy), (0, 255, 0), 5)
cv.putText(frame, "Center", (cx - 20, cy - 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

# Display the resulting frame
cv.imshow(" Webcam", frame)

# Update maximum area
if area > ms:
    ms = area
    print("Area:", ms)

# Send data to Arduino
float_var = ms
ser.write(struct.pack('f', float_var))
time.sleep(1) # Wait for Arduino to process data

# Wait for 1 second
time.sleep(1)

# Increment loop counter
ms += 1

# Check for key press; if 'q' is pressed, exit the loop
if cv.waitKey(1) & amp; not == ord('q'):
    break

# Close serial port connection
ser.close()

# Release the video capture object and close all windows
cap.release()
cv.destroyAllWindows()

```

2

Код по классификации мусора и отправки данных на Arduino

```

1 from keras.models import load_model
2 import cv2
3 import numpy as np
4 import serial
5 import time
6
7 np.set_printoptions(suppress=True)
8
9 # Загрузка модели
10 model = load_model("keras_model.h5", compile=True)
11
12 # Загрузка файла labels.txt
13 class_names = open("labels.txt", "r").readlines()
14
15 # Подключение к Arduino
16 arduino = serial.Serial('/dev/ttyUSB0', 9600)
17
18 url = "http://172.20.10.6:4242/video"
19 camera = cv2.VideoCapture(url)
20
21 # Переменная для хранения имени
22 last_send_time = time.time()
23
24 while True:
25     ret, image = camera.read()
26
27     # Изменяем размер изображения до (224x224) пикселей
28     image = cv2.resize(image, (224, 224), interpolation=cv2.INTER_AREA)
29
30     # Преобразуем изображение в формат
31     cv2.imshow("Webcam Image", image)
32
33     image = np.asarray(image, dtype=np.float32).reshape(1, 224, 224, 3)
34
35     image = image / 127.5 - 1
36
37     # Выводим прогноз и уровень уверенности
38     prediction = model.predict(image)
39     index = np.argmax(prediction)
40     class_name = class_names[index]
41     confidence_score = prediction[0][index]

```

1

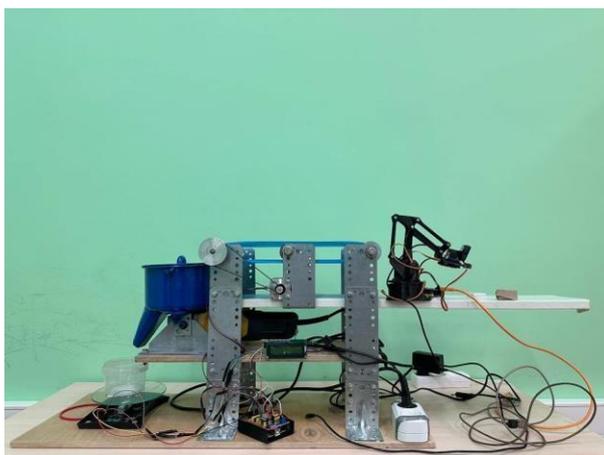
```

45 # Выводим прогноз и уровень уверенности
46 print("Класс:", class_name[2], end="")
47 print("Уровень уверенности:", str(np.round(confidence_score * 100))[:-2], "%")
48
49 # Отправляем данные на Arduino каждые 10 секунд
50 if time.time() - last_send_time >= 10:
51     last_send_time = time.time() # Обновляем время последней отправки
52     if "plastic" in class_name:
53         arduino.write(b'10') # Отправляем '10' для пластика
54     elif "paper" in class_name:
55         arduino.write(b'20') # Отправляем '20' для бумаги
56     else:
57         # Отправляем значение по умолчанию (например, 0), если не пластик и не бумага
58         arduino.write(b'0')
59
60 # Слушаем нажатия клавиш на клавиатуре.
61 keyboard_input = cv2.waitKey(1)
62 # 27 - это ASCII-код клавиши Esc на клавиатуре.
63 if keyboard_input == 27:
64     break
65
66 camera.release()
67 cv2.destroyAllWindows()
68

```

2

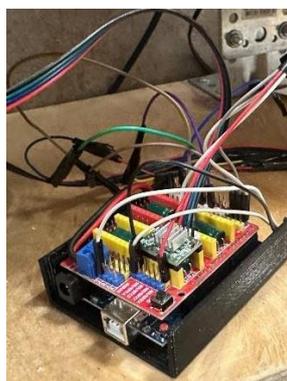
Фотографии элементов имитационной модель:



Имитационная модель



Дисплей



Плата cnc shield v3



Конвейер



Камера esp32-cam

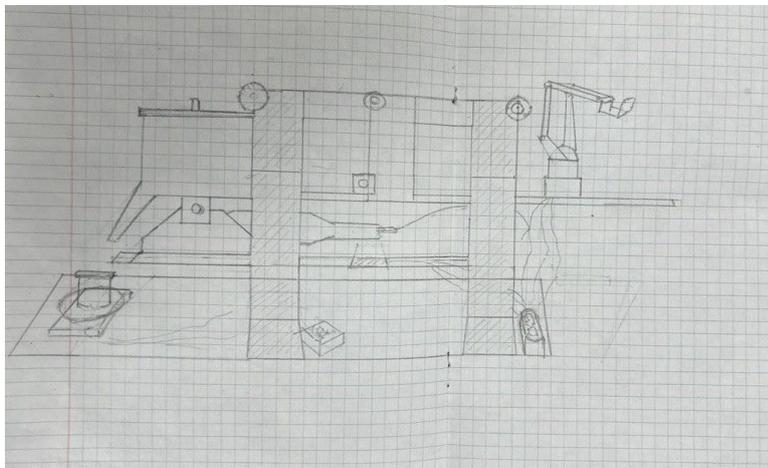
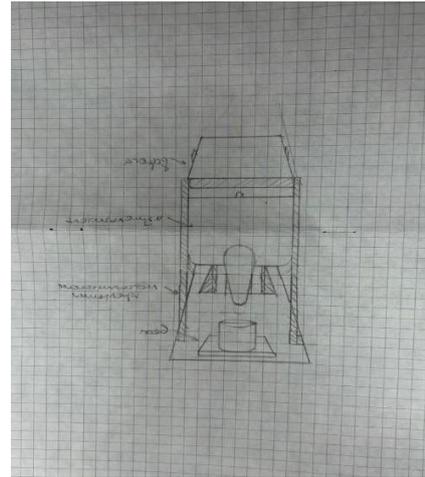
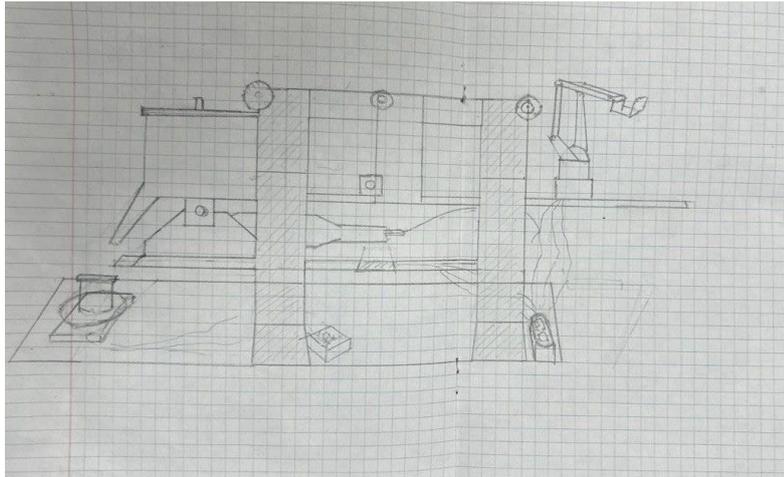


Измельчитель

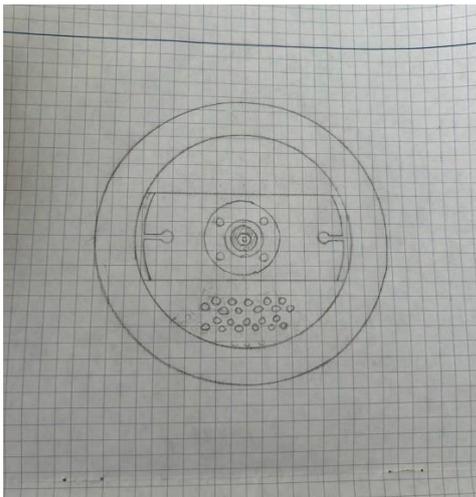


Манипулятор

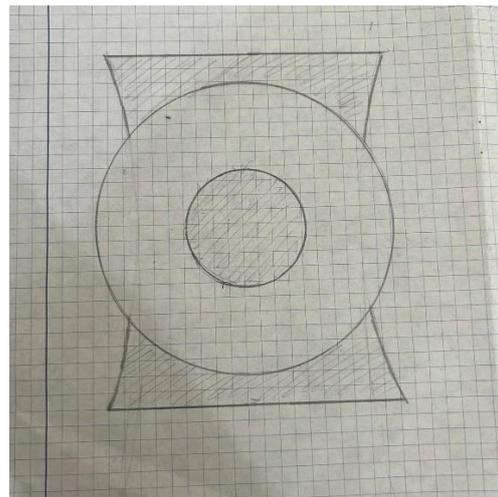
Эскизы имитационной модели:



Эскизы отдельных частей имитационной модели:



Измельчитель



Весы

