

Проект “ГВИНТИКИ”

(винтики)

Предисловие

Мне было интересно попробовать реализовать проект не только со стороны программирования, а и технической стороны, решить полностью “проблему”, чтобы в результате получить готовый продукт который поможет закрыть потребность.

Проблема

У нас большой набор спинальных винтов для лечения позвоночника. Их определенное количество видов и на складе мы их распределяем и храним по артикульно. К сожалению, на склад может приходить набор разных винтов в одном контейнере и работнику склада необходимо их разделить по артикульно, для этого необходимо обучать, чтобы понимал чем отличается один от другого, бывают различия минимальны.

Задача

Детекция объектов и распознавания
(классификация).

Разработка решения для удобства
подготовки данных для обучения.

Создание “пространства” для
использования результата.

Внедрение решения.

Пример данных



Пример реальных винтов.

Дальше для прототипа системы мы использовали не медицинские винты, так как после любой "фотосессии" винтов необходимо их отправлять на стерилизацию и заново упаковать. Для проекта который был придуман "а получится ли" слишком дорого выводит постоянно стерилизовать 😅

И вот “гвинтики” для проекта



Этап 0: Начало

Подготовка места

Придумать концепт, как удобно получать данные.

“На коленке” быстро собрали вот такой стенд.

Штатив, закреплен проектор. рядом крепление для фотоаппарата.



Этап 0: Начало

Подготовка места

Проектор на первых этапах используем как для подсветки всего листа и в дальнейшем реализуем что результат скрипта это обвести квадратом объект и подписать что это (артикул например).

Пример фото со стенда

----->



Этап 1: Подготовка данных

Концепт

Найти лист на фото и вырезать его.

Найти винты на фото.

Обучить нейронку.

Этап 1: Подготовка данных

Найти лист

Отличный 6 и 7 урок который
должен решить эту задачу. Но
столкнулся с проблемой.

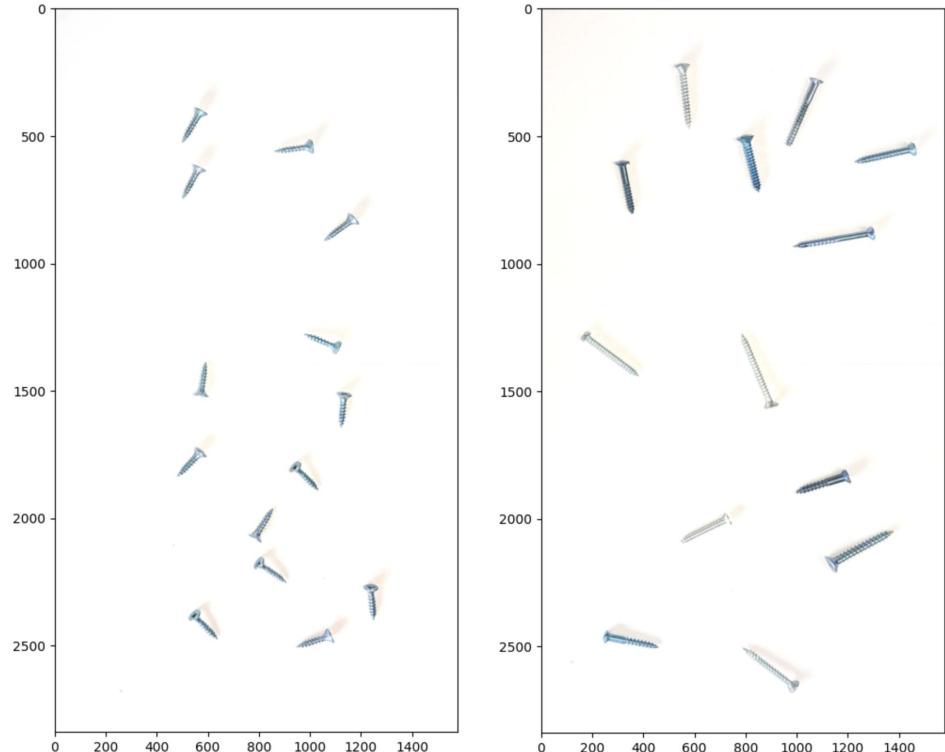
Все никак не мог понять почему
часто углы не верно находит и
этот процесс занимает +- 10-15
секунд.



Этап 1: Подготовка данных

Найти лист

Другое решение (урок 4). Найти все прямые и потом их кластеризовать и найти пересечения на углах и вырезать картинку.
Получилось!



Этап 1: Подготовка данных

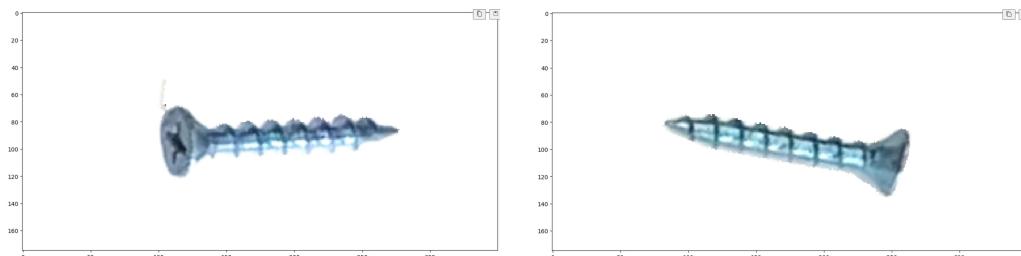
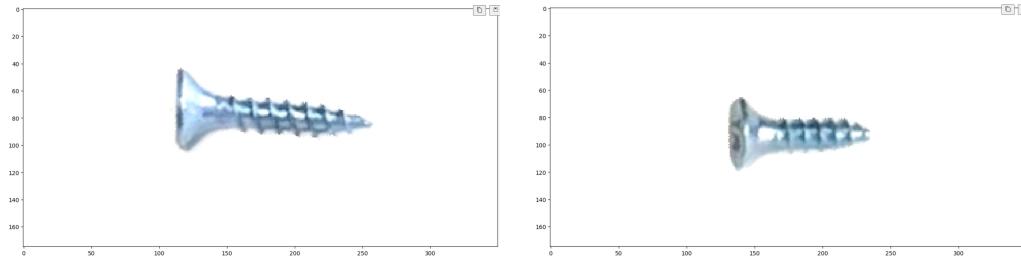
Найти винты

findContours... GaussianBlur...

findContours... GaussianBlur...

findContours... GaussianBlur...

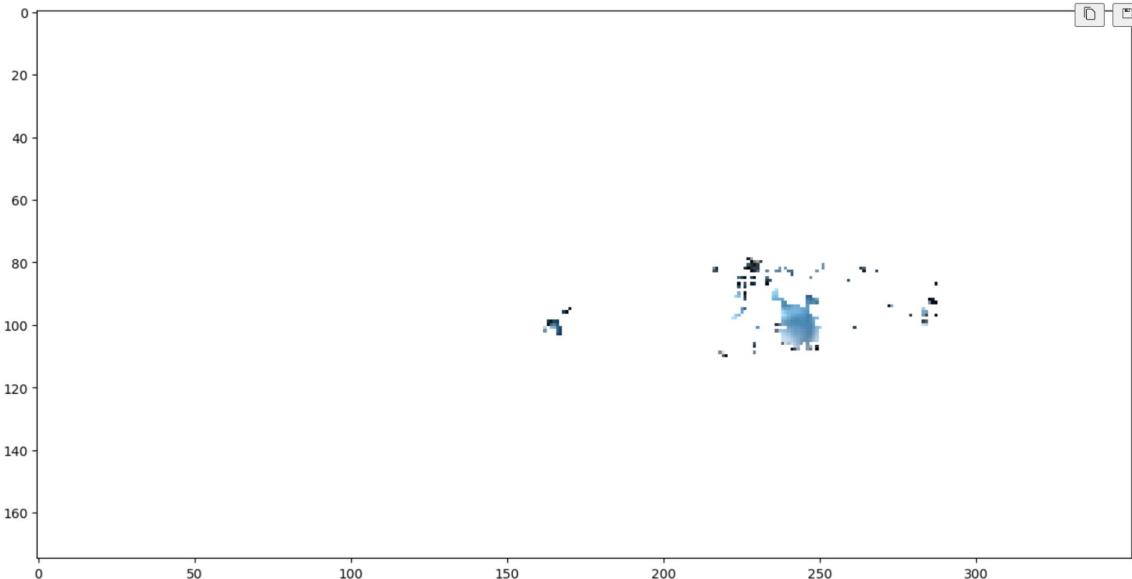
GaussianBlur необходим чтобы исключить мусор. Свет не равномерный и он какие-то области принимал как контур.



Этап 1: Подготовка данных

Найти винты

но есть и недочет этой
системы...



Этап 2: Нейронка

Обучение

Решили попробовать с Inception v3. Пробовали много разных подходов. Последнее попытка была замораживали слои которые отвечают за распознавание простых элементов (круги/линии) - это макспулинги и коволюции до слоя mixed7 и добавить слои Dense и Dropout и доучить уже “плотные” слои.

jpeg	IMG_08398.jpeg
true_class	14
predict_class	10
confidence	43
correct_class	False
condition	TN
Name:	55, dtype: object

jpeg	IMG_08392.jpeg
true_class	14
predict_class	13
confidence	63
correct_class	False
condition	TN
Name:	86, dtype: object

jpeg	IMG_07769.jpeg
true_class	1
predict_class	1
confidence	48
correct_class	True
condition	FN
Name:	84, dtype: object

jpeg	IMG_07807.jpeg
true_class	3
predict_class	3
confidence	56
correct_class	True
condition	FN
Name:	59, dtype: object

jpeg	IMG_07766.jpeg
true_class	1
predict_class	1
confidence	50
correct_class	True
condition	FN
Name:	20, dtype: object

jpeg	IMG_07764.jpeg
true_class	1
predict_class	5
confidence	56
correct_class	False
condition	TN
Name:	94, dtype: object

Этап 2: Нейронка

Обучение

Результат был не на “высоте”,
но это и понятно, у нас было 15
классов и в каждом в среднем
20 изображений. Не так много
данных для обучения.

```
jpeg      IMG_08398.jpeg
true_class          14
predict_class        10
confidence           43
correct_class        False
condition            TN
Name: 55, dtype: object
```

```
jpeg      IMG_07769.jpeg
true_class          1
predict_class        1
confidence           48
correct_class        True
condition            FN
Name: 84, dtype: object
```

```
jpeg      IMG_07766.jpeg
true_class          1
predict_class        1
confidence           50
correct_class        True
condition            FN
Name: 20, dtype: object
```

```
jpeg      IMG_08392.jpeg
true_class          14
predict_class        13
confidence           63
correct_class        False
condition            TN
Name: 86, dtype: object
```

```
jpeg      IMG_07807.jpeg
true_class          3
predict_class        3
confidence           56
correct_class        True
condition            FN
Name: 59, dtype: object
```

```
jpeg      IMG_07764.jpeg
true_class          1
predict_class        5
confidence           56
correct_class        False
condition            TN
Name: 94, dtype: object
```

Что дальше...

Работа над ошибками

Пересмотреть некоторые участки логики.
Например нахождение листа на фото.

Попробовать другой подход в обучении (урок 18)

Аннотации...

Нахождение листа

Внимание к деталям

Первый раз когда я искал углы с помощью Harris Corner Detector я не обратил внимание что у меня картинка размером 4000 на 3000 пикселей.

и тут я понял...

resize до 800 на 600 и результат отличный, все логика отрабатывает за 1-2 секунды. и с помощью getPerspectiveTransform вырезаем лист



ДО



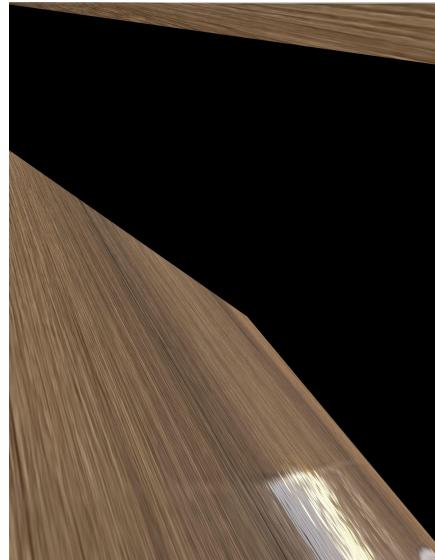
ПОСЛЕ

Нахождение листа

Внимание к деталям



← результат



но бывает и
такое

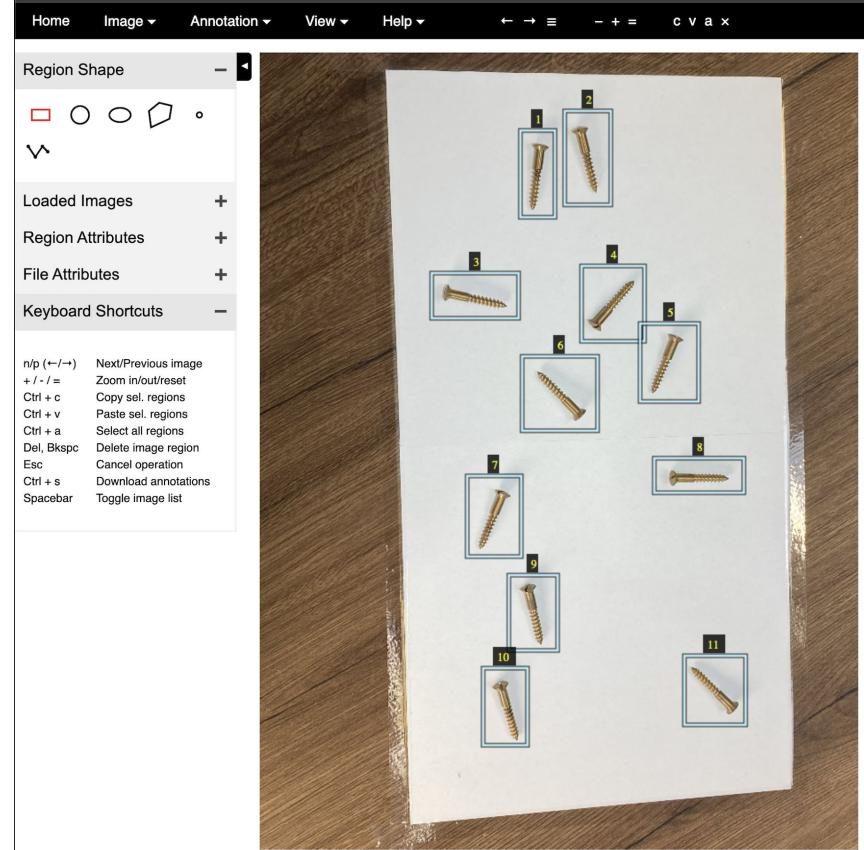
С таким результатом
столкнулся только пару
раз, автоматизированное
решение не делал, просто
руками исключил эти
фото.

В планах после
трансформации добавить
проверку на количество
“белых” пикселей и если
не достаточно, то
исключать такое
изображение.

Урок 18

Yolo

После просмотра 18 урока я понял что можно попробовать решить задачу не обрабатывая изображения. А сделав к ним аннотации. Используя самый простой инструмент что нашел <https://annotate.officialstatistics.org/> начал делать аннотации к последней “фотосессии” из 65 фото. Решил попробовать просто все винты выделить не классифицируя.



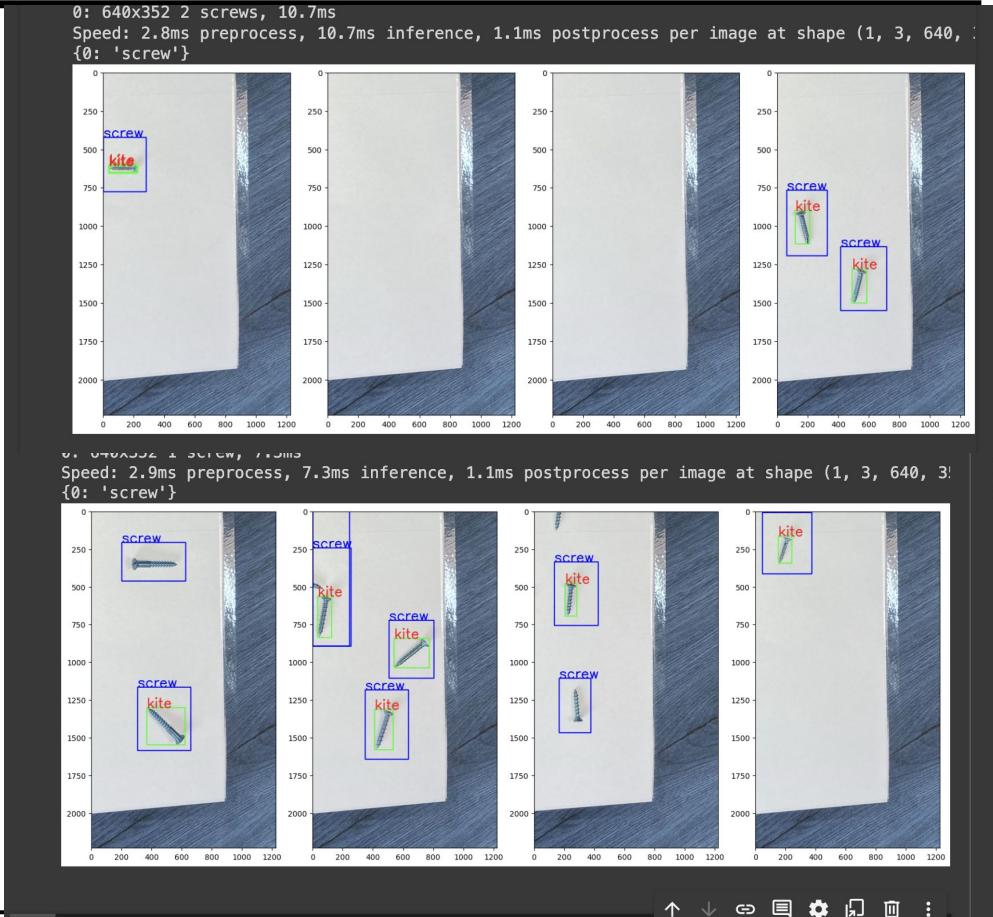
Урок 18

Yolo

После подготовки данных и обучения нейронки на yolov8n получил вот такой результат на 10 эпохах.

Вроде хорошо, но дальше окажется не так...

P.S. yolov8n всегда определяет их как воздушный змей, смешно :)



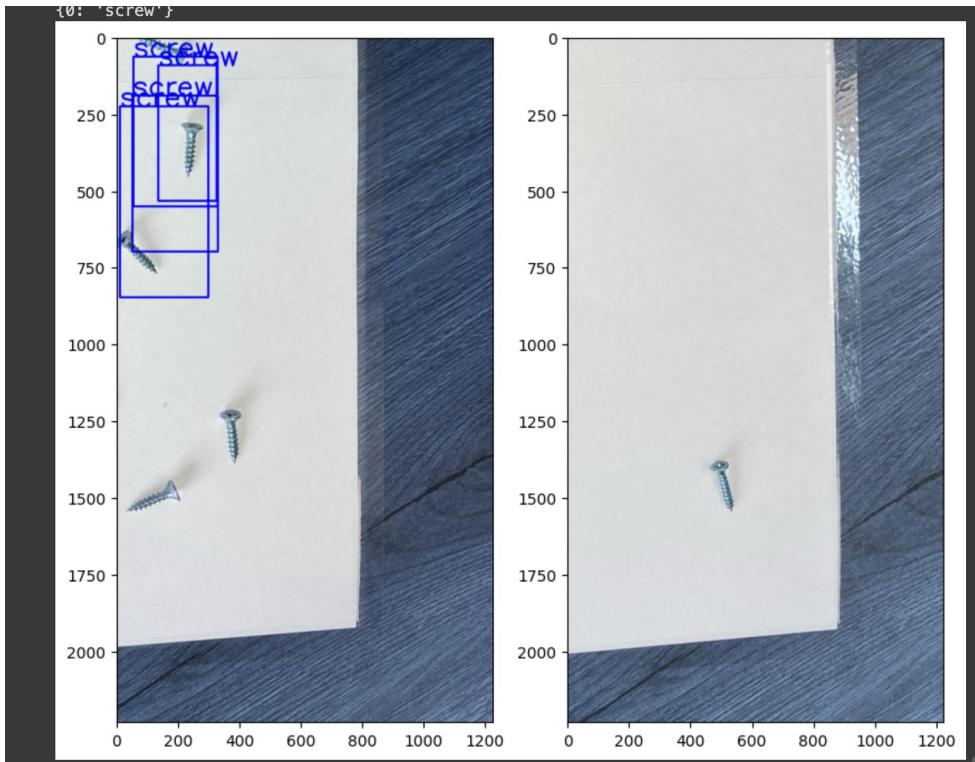
Урок 18

Yolo

Вот тут видим что очень плохо все.

Проведя в последствии анализ я понял что аннотации, все таки, надо делать точнее, не оставлять много отступов.

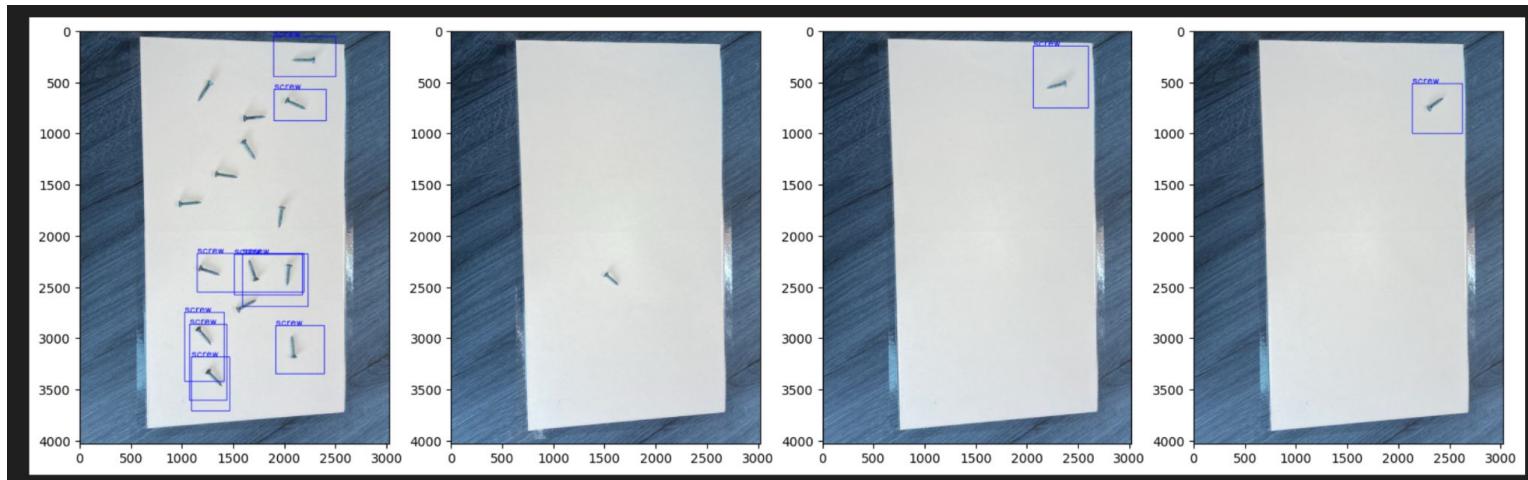
Попробовал запустить с этими же аннотациями, но 100 эпох...



Урок 18

Yolo

Но результат не поменялся. Все упирается в точность аннотаций и количество данных для тренировки. Тут использовалось 65 фото и 316 аннотаций к ним. Что достаточно мало.



Что дальше... (x2)

Далее по плану

На сейчас из практической части это все что реализовал. И вот какой план дальше.

Данные для обучения

Реализовать стенд для полуавтоматического сбора данных для обучения неронки.
Использовать автоматическое обнаружение винтов что ранее делал для того чтобы
подготовить аннотации, конечно с возможностью исправления.

Интерфейс

Сделать интерфейс для обработки данных.

Предподготовка

Всетаки решил вернуться к тому что необходимо изображение подготовить. Так как это
задумка в виде стандарта, то есть на проде будут те же условия, то можем попробовать
кучу лишних данных откинуть.

Далее по плану

Регуляризация

В своей работе я это не использовал, попробую добавить, скорее всего это позитивно отразится на обучении модели

Другие нейронки

Попробовать YOLOv8 OBB

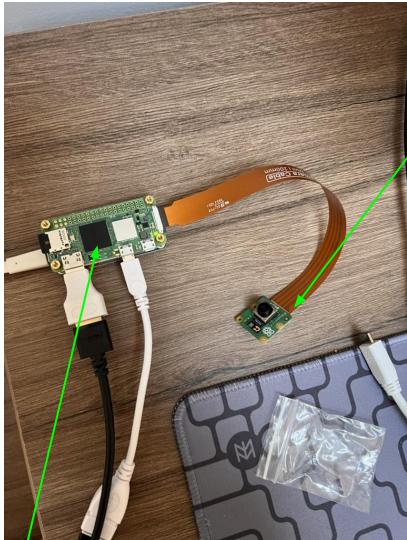
Вывод результата

Тривиальная задача. Вывести квадратики не на дисплей монитора, а на проектор который будет выделять на столе винт.

Внедрение

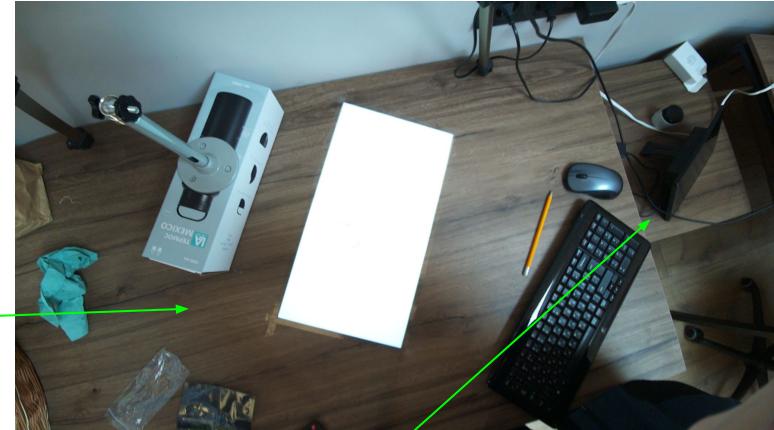
Попробовать внедрить в работу :)

План выполняется...



rpi camera model 3

пример фото с
камеры
(чуть смазал, так как
камеру держал руками)



а это стоит 7" тач дисплей в который
вставляется rpi, на нем начал делать
интерфейс для уже финального продукта.
чтобы можно было обходиться без ноута.
Например на нем можно сейчас тыкнуть на
кнопку которая отправляет команду другой rpi
сделать фото, и она отправляет уже дальше в
pipeline фото.

rpi zero w v2

**Спасибо за
внимание!**

A —

