

Ведрусоход - ver.1

Ведросоход v.2.2

15 декабря 2024 начал второй сезон. За лето наигравшись в моряка, повернул голову направо ... ба! Там робот стоит, пылится. Ждёт меня, поигрывая на виду новыми энкодерами!

План на 2025+

Размещение компа на платформе:

- снимаю верхнюю фанерку
- лезу под фанерку и решаю проблемы BMS
 - меняю на новый BMS и тестирую её датчик тока
 - паяю два разъёма XT-60 и вывожу их более назад, чем щас
- креплю короба ПОД фанерку
- ставлю фанерку на место
- формирую виброзащиту компу
- кидаю комп посрединке на виброзащиту
- креплю всё ранее оторванное
 - драйверы моторов
 - вперёд, соответственно, креплю первую stm32 для управления моторами. Оставляю место под её max3232
 - выключатели питания с учётом макета dc/dc
 - ? боковые сонары
 - нормальные разъёмы на длинных проводах, проложенных в коробах

n8n и / или LLM

TBD

Интересный конструктор.

- краткий рассказ, что за n8n такой. Примеры собранных на нём решений;
- моя мелкая сборка на локальной LLM
 - с мелкой моделью (видео ускорено, но счётчик секунд заметный);
 - с моделью побольше (типа, умнее, но сильно тормознее);
- openai

Проверка на земле

- разобраться с коррекцией камер и 360 обзором в YOLO
- тестить поворот к персоне дома и движение за персоной на улице

360° Depth

1. MiDaS (Monocular Depth Estimation)

- **Ссылка:** github.com/isl-org/MiDaS
- **Актуальность и поддерживаемость:**
Активно поддерживается группой Intel ISL, регулярно обновляется, имеет большую базу пользователей и применений в научных исследованиях.
- **Производительность:**
Высокая точность при оценке глубины; на современных GPU модель способна работать в реальном времени (при оптимизации и подходящей аппаратуре).
- **Описание:**
Современная модель, использующая нейросети (включая трансформерные архитектуры) для получения детальной карты глубины по одному изображению. Идеально подходит для сложных сцен и не требует стереозрения.

2. MonoDepth2

- **Ссылка:** github.com/nianticlabs/monodepth2
- **Актуальность и поддерживаемость:**
Проект от Niantic Labs, активно используется сообществом. Хотя основные обновления могут идти не так часто, модель остаётся надёжной и актуальной.
- **Производительность:**
Хорошее соотношение между точностью и скоростью; оптимизирована для работы в реальном времени на GPU и адаптирована для мобильных платформ.
- **Описание:**
Самоконтролируемая модель для оценки глубины с одного изображения, использующая архитектуру энкодер-декодер. Обучается без необходимости в полностью размеченных данных, что упрощает применение в DIY проектах.

3. AdaBins

- **Ссылка:** github.com/shariqfarooq123/AdaBins
- **Актуальность и поддерживаемость:**
Современное решение с активной поддержкой сообщества; проект регулярно получает обновления и демонстрации результатов.
- **Производительность:**
Отличается высокой детализацией предсказаний, но для работы в реальном времени требует GPU, так как модель несколько вычислительно тяжёлая.
- **Описание:**
Модель использует адаптивное разбиение глубинного пространства, что позволяет

получать более точные и детализированные глубинные карты даже для сложных сцен.

4. DPT (Dense Prediction Transformer)

- **Ссылка:** github.com/isl-org/DPT
- **Актуальность и поддерживаемость:**
Разработан тем же коллективом, что и MiDaS, интегрирован в их экосистему; регулярно обновляется и активно используется.
- **Производительность:**
Высокая точность за счёт использования трансформерных архитектур; требует современных GPU для оптимальной работы.
- **Описание:**
Модель использует возможности трансформеров для создания детальных карт глубины, обеспечивая высокое качество даже при неоднородном освещении и сложной структуре сцены.

5. BTS (Beyond The Surface)

- **Ссылка:** github.com/cleinc/bts
- **Актуальность и поддерживаемость:**
Проект опубликован в научной литературе и имеет поддержку в исследовательском сообществе. Обновления происходят, хотя активность может варьироваться.
- **Производительность:**
Демонстрирует конкурентоспособную точность и в некоторых случаях может работать в реальном времени на GPU.
- **Описание:**
Модель ориентирована на восстановление детализированных структур сцены с одного изображения, что делает её подходящей для задач робототехники, AR и других приложений, где важна точность глубинной информации.

6. FastDepth

- **Ссылка:** github.com/dwofk/fast-depth
- **Актуальность и поддерживаемость:**
Проект ориентирован на встраиваемые и мобильные системы; имеет активное сообщество пользователей, особенно в сфере робототехники.
- **Производительность:**
Очень быстрая модель, оптимизированная для работы на ARM-платформах и NVIDIA Jetson, что позволяет использовать её в реальном времени на ограниченных вычислительных ресурсах.

- **Описание:**

Легковесное решение для оценки глубины, разработанное с учётом требований встраиваемых устройств. Обеспечивает быстрые предсказания, что делает её идеальной для DIY роботов с USB камерами.

Порядок в проводах

На этапе подключения периферии куча проводов стали висеть в воздухе. Надо всё это прибрать:

- разложить какие-то аля-кабель-каналы (или другие короткие защёлки), прибрать туда (удлиненные) провода к сонарам, датчикам тока и так далее. Не хочу термоклеить шнуры - хрен достанешь потом;

TBD

микрофон, динамик. На реле питание моторов. Пусть мозги отключают их, когда батареек мало. Получениеnapруги аккумуляторов или через BMS, или через АЦП.

YOLO разные потестить

Этих YOLO всё больше (а ещё RT-DETR есть). Надо понять что есть что

- yolo-nas
<https://docs.ultralytics.com/ru/models/yolo-nas/#how-can-i-integrate-yolo-nas-models-into-my-python-application>
шустрый и точный?
- pp-yolo
<https://github.com/PaddlePaddle/PaddleDetection/blob/release/2.8.1/configs/ppyolo/README.md>
- yoloe <https://github.com/THU-MIG/yoloe>
Как я понял, быстрый yolo-world

BMS

Тестировать новую плату

- датчик тока
- Bluetooth
- UART

Я пытался подключить её во время подключения остальной периферии. Но то ли зуб в апельсине гнилой, то ли (скорей всего) софт не движется. Надо попробовать на новой флешке / ОСи сконнектиться, чтобы закрыть железный вопрос. Или новый BMS заработает на этом софте или UART (не вижу питания на UART)

- во-вторых, в этом апельсине синий зуб есть? Чтобы обойтись без проводов.
- и что за софт нужен? Мне было б круто питон-модуль запустить. Кажется, этот: <https://github.com/fkressmann/python-jbdtool-bms>

А лучше даже сразу ROS-ноту готовую!

https://github.com/RobotnikAutomation/jbd_bms
https://github.com/I-Quotient-Robotics/jbd_bms_ros

- <https://github.com/robwolff3/jbd2mqtt> тут пишут про UART, хоть софт не совсем мне нужный
- <https://github.com/sshoecraft/jbdtool> старая софтина. Собирается только со старой gattlib
- pip3 install ble-serial или <https://github.com/Jakeler/ble-serial> хороша. Но и она не подключилась к BMS. Сбрасывать BMS и комп пробовал
- orangepi-config / network / BT discover тоже не находит ничего

Усложнение режимов

Как только закончу тестирование режима следования за персоной с подъездом к ней, так начинаю закладывать основу картографии и делать интеграцию с LLM, видимо.

Парктроник от машины

Чёт типа этого: <https://aliexpress.ru/item/1005004276888072.html>

Parking Sensor Kit



Вот его (не полный) реверс

- <https://www.electroschematics.com/secrets-of-car-reverse-parking-sensors/>
- <https://www.candrian.gr/index.php/reverse-engineering-car-parking-sensors/>

На будущее

Предполагаемые векторы использования нейросетей в проекте моего робота:

- обнаружение и распознавание объектов с помощью YOLO, YOLO-world и LLM-VL;
- сегментация изображения, построение карты глубины со всех камер с помощью [Depth Anything](#) и [FastDepth](#);
- обработка естественного языка с помощью Whisper, Vosk и LLM;
- использование трансформеров для предсказания временных рядов данных

- сенсоров и векторов движения окружающих объектов;
- RGBD SLAM с использованием [Droid SLAM](#) и [Nice SLAM](#);
- управление жестами (YOLO, [Mediapipe](#))

Стереозрение

Используя OpenCV и пару веб-камер, направленных вперёд, смогу видеть глубину.

Изучить:

- <https://habr.com/ru/articles/446872/>
<https://github.com/realizator/stereopi-tutorial>
- <https://waksoft.susu.ru/2021/01/26/sozдание-nedorogoj-stereokamery-s-ispolzovaniem-opencv/>
- <https://habr.com/ru/articles/388259/>
- https://github.com/raulmur/ORB_SLAM2
- https://docs.opencv.org/4.x/dd/d53/tutorial_py_depthmap.html
- ROS Stereo
 - http://wiki.ros.org/stereo_image_proc
 - http://wiki.ros.org/stereo_image_proc/Tutorials/ChoosingGoodStereoParameters

!! Что, конечно же, полезно для определения препятствий в будущих версиях !!

Сбор датасета участка

- научить бота ездить в FPV режиме с трансляцией по wifi всех камер на монитор(ы) домой;
- организовать запись изображений со всех камер;
- по всем изображениям дома, на хорошей видяхе, пройти через YOLO World и найти нужные мне объекты;
- сконвертировать картинки и объекты в датасет, подходящий для COCO anotator или, лучше, roboflow;
- пройти по всем картинкам руками, отфильтровав мусор;
- использовать датасет для обучения YOLOv8.

Обновить ROS2

Чёт тут пишут, что мой IRON уже устарел... Да и пятое ведро линуха помойное уже (хоть к нему у меня rockchip-tools привяз).

С другой стороны, если в перспективе я перехожу (или добавляю) на amd64, то от апельсина нужна лишь двусторонняя прокса к железу. И эту функцию я бы перенёс на большой комп. Не знаю пока, как именно. И стоит ли.

Структурированный ответ от LLM

https://github.com/abdallah-ali-abdallah/pydantic-ai-agents-tutorial/blob/main/2_simple_structure_d.py, включая анализ картинки

Анализ глубины из обычных камер

А не будет ли интересно что-то подобное замутить на задних камерах? Конечно же, NPU рокчиповский вряд ли будет помогать. Но ... интересно!

- git clone <https://github.com/DepthAnything/Video-Depth-Anything>
- cd Video-Depth-Anything
- pip install -r requirements.txt

Анализ, что сейчас вижу на снимке

<https://huggingface.co/nvidia/Eagle2-9B>



What constellation is above the house?

The constellation above the house is Orion, but the image is too low resolution to make out the nose. The constellation consists of three stars aligned in a row and three more stars above, creating a pattern resemblance of an hourglass for when being viewed from a side. The two stars that make up Orion's 'belt' are not visible in the image. Overall, this picture is probably taken at around 23:50 UT, which creates a very "Orion Rich" sky, displaying the glowing

Камеры вокруг

В перспективе ... остальные камеры гавно! Хочу:

- usb3 или езер, явно не хватает usb2 скорости
(в принципе и чёрт с ним пока. Щас есть 3 x 800x600, что подходит к 640x640)
- rgb24
(в принципе и чёрт с ним пока. Щас 422. То есть потери в цвете)
- ширик
(да, но и щас шире некуда)
- автояркость лучше
(в принципе, могу хоть сам подбирать выдержку. Я всё равно кадры смотрю, правлю. Могу и выдержку менять)
- fps 5
(а оно и щас так)

VFH (Vector Field Histogram)

VFH (Vector Field Histogram) — это алгоритм навигации для мобильных роботов, который позволяет избегать препятствий и двигаться к цели. Он широко используется в

робототехнике, особенно в сочетании с лидарами или камерами глубины, которые предоставляют информацию о расстоянии до объектов в окружающей среде.

Основные принципы VFH:

1. Построение гистограммы препятствий:

- Лидар или камера глубины сканирует окружающую среду, предоставляя данные о расстоянии до объектов.
- Эти данные преобразуются в гистограмму, которая представляет собой круговой массив, разделенный на сектора (например, 360 секторов для полного круга).
- Каждый сектор содержит информацию о плотности препятствий в данном направлении.

2. Анализ гистограммы:

- Алгоритм анализирует гистограмму, чтобы определить "опасные" направления, где плотность препятствий высока.
- На основе этого анализа строится векторное поле, которое указывает, в каких направлениях робот может безопасно двигаться.

3. Выбор направления движения:

- Робот выбирает направление, которое одновременно:
 - Близко к направлению на цель.
 - Свободно от препятствий.
- Если прямое движение к цели невозможно из-за препятствий, робот выбирает обходное направление, которое минимизирует отклонение от цели.

4. Корректировка траектории:

- По мере движения робот постоянно обновляет гистограмму и корректирует траекторию, чтобы избегать новых препятствий.

Преимущества VFH:

- **Реактивность:** Алгоритм быстро реагирует на изменения в окружающей среде.
- **Простота реализации:** VFH относительно прост в реализации и требует небольших вычислительных ресурсов.
- **Гибкость:** Может работать с различными типами датчиков (лидар, камеры глубины, ультразвуковые датчики).

Ограничения VFH:

- **Локальные минимумы:** В сложных средах робот может застрять в локальных минимумах, если препятствия окружают его со всех сторон.

- **Ограниченная дальность планирования:** VFH ориентирован на ближайшие препятствия и не учитывает глобальную карту, что может привести к неоптимальным маршрутам.

Применение:

VFH используется в роботах для:

- Навигации в динамических средах (например, склады, офисы).
- Избегания столкновений с людьми или другими роботами.
- Локального планирования траектории в сочетании с глобальными алгоритмами (например, A* или RRT).

Пример работы:

1. Робот сканирует окружение лидаром и строит гистограмму.
2. Алгоритм определяет, что прямое движение к цели заблокировано препятствием.
3. Робот выбирает направление влево или вправо, где плотность препятствий ниже.
4. Робот движется в выбранном направлении, постоянно обновляя гистограмму и корректируя траекторию.

Таким образом, VFH — это эффективный инструмент для локальной навигации, который позволяет роботу безопасно двигаться в сложных и динамических средах.

Задачи ведрусохода

Живу я в СНТ “Ведрусия”. Робот, в перспективе, будет перемещаться по гектарному заросшему участку (отличный полигон!). Отсюда его название. Типа, луноход ходит по Луне. Марсоход ходит по Марсу. Ведрусоход же, само собой, по Ведрусии :)

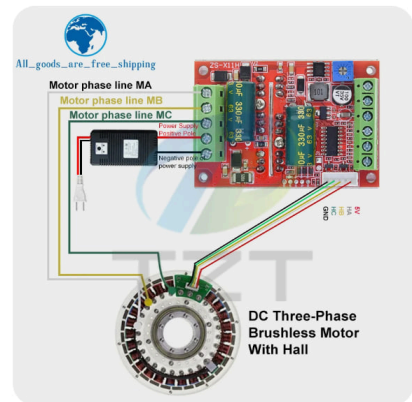
Важно держать в голове при проектировании робота, что это не робот-аватар. Это именно автономный комплекс, выполняющий текстовые и голосовые команды оператора.

Принимающий решения на основе текущего статуса, команд, анализа изображения нейросетями и показаний датчиков.

Ведрусоход - робот эволюционный. Его (и мой) опыт, навыки и умения растут день ото дня. Сейчас нет чёткой дорожной карты. Я просто иду вперёд и надеюсь на лучшее!

Тележка

Механика



Мелкий бот ходит на двух 10" мотор-колёсах от гироскутера. Раз два мотора выдерживают мои 80 кило, то точно выдержат ~20 кило аккумуляторов, кило плат и допы 😊

Механика, для начала, простая. Колёса от гироскутера без переделки - на полуосях. Рама сварена из 20x40 профтрубы с 20x20 перемычками. Колёса приболтованы стяжками для троса d16. Пока что не думаю о влагозащите. Но, в перспективе, герметичный всепогодный бот.

Хард (электроника)

Управление моторами независимое - через два (гавёных) 60v контроллера с али.

Моторы на скутере питаются от 36 вольт, вот и не стоит их перегружать. Для начала, из Leaf модулей собрал 10s вольт батарею, добавил Smart BMS. При ограничении тока контроллера в 15 ампер получаю по пол киловатта на мотор или 1 кВт на тележку. Более чем для такого малыша. Контроллеры заявлены 200...350 Вт, то есть около 10 ампер штатно.

На входе питания каждого контроллера - датчик тока. Так можно понять / отрегулировать усилие на колесе.

Снаружи каждого мотор-колеса стоят инкрементальные энкодеры на 1000 тиков помножить на редуктор 4 к 1.

STM32 BluePill две. Одна на левое, вторая на правое колесо. Можно было б и одну. Может и стоит одну. Но пока так.

К апельсину подключена периферия:

- I2C: BME280
- I2C: ADS1115 (АЦП на датчики тока)

- GPIO сонары
- набор мосфетов / реле на грелку подкупольного, прожектор и другие бинарные нагрузки;
- I2C ШИМ демультимплексор на 16 каналов. Сервы, пока для баловства. Потом и для дела (рука, нога, чего там).

Мозги

Мозг робота - новый апельсин-3В. Мощная платка, заказал одну на пробу:

- arm64 проц;
- 8 гиг мозгов!
- встроенный wifi 5;
- eMMC модуль (я заказал на 64 гиг, есть и больше);
- type-C питание;
- NPU для YOLO и прочих тензоров (сходу хз как задействовать);
- h264/h265 железный кодер (сходу хз как задействовать);
- аудио вход / выход под колонки / микрофон;
- 4 USB для вебок / ардуинок (маловато .. у меня две вебки вперёд, пара назад и две арду. Пассивный хаб понадобится);
- порт под 5В вентилятор.

Задачи мозга первой версии

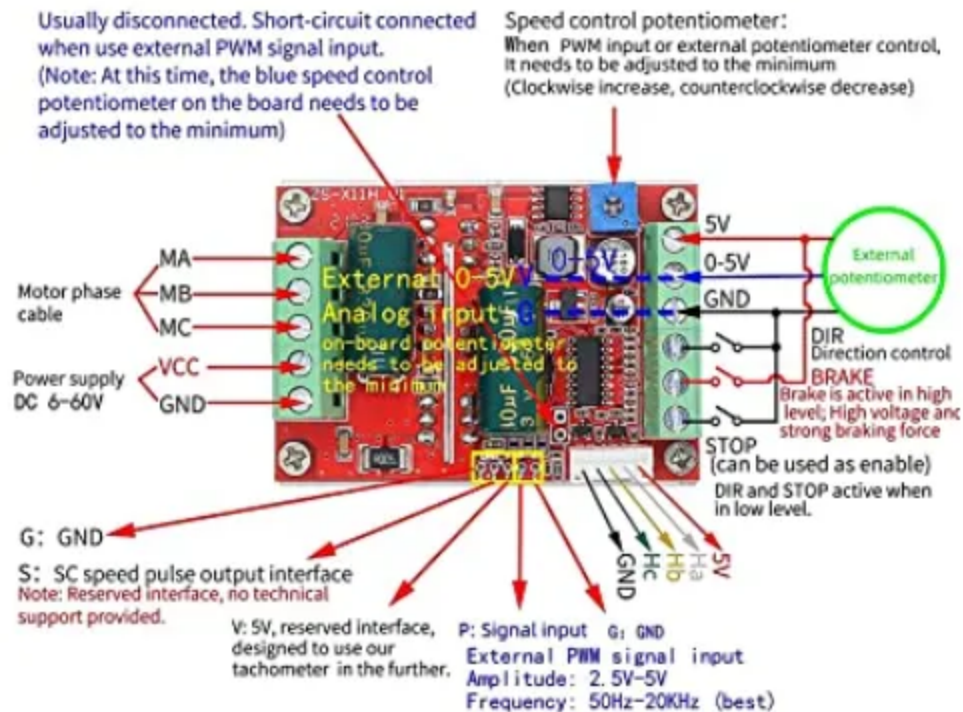
Ставлю ROS, разбираюсь.

Технические задачи:

- SLAM (создание и обновление карты) с двух передних вебок (стереокамера);
- YOLO со всех камер (две передние и две задние);
- RGBD (стереокамера) с двух передних. Чтобы YOLO в итоге говорила и кто / что, и где, и как далеко.

Поведенческие задачи:

- шлёт тревоги в телегу:
 - детектор YOLO нашёл человека / машину / животное;
 - детектор шума (собачий лай, шум машины, голос людей. Хорошо бы с направленным микрофоном понять где шумит);
 - малый заряд АКБ;
 - застрял (колесо не крутится).
- перемещение по карте;
- засада (охрана на месте):
 - в текущем месте сидит на месте;
 - читает 4 вебки, детектирует окружение через YOLO;
 - шлёт тревоги.



Продумать

- ODrive - [FOC контроллер мотор-колеса на али](#)
- [slam "yolo"](#) - Поиск в Google
- https://youtu.be/4OfCKg9vSVc?si=c70N_fJ9iKhDDtvj - RGBD для SLAM!
- **ДОРАБОТАТЬ** Этап: "веб сервер и пульт". Часть 3, "функционал дополнительный" [??/??/2024 - ??/??/2024]
 - Сценарий:
 - Есть каркас пульта. Для успешной настройки бота нужно развить функционал:
 - блок на запуск без пульта, контроль keepralive пульта;
 - установка параметров:
 - ardu:
 - max_power;
 - controller:
 - ROTATE_SPEED_MAX;
 - PID поворота: p, i, d.
 - управление:
 - STOP (авария);
 - restart контроллера по изменению его параметров.
 - Софты:
 - server

- шлём *keepalive* пульта по получению */status* запросов;
 - *index.html*
 - форма установки параметров.
 - *controller*:
 - установка;
 - сброс в ноль
 - **STOP** кнопка.
 - *controller*
 - блок на запуск без *keepalive* пульта (как сейчас ждёт *keepalive safety*);
 - проверка *keepalive* пульта (гар 10 сек). Авария!
 - подписка на изменение параметров от *server*. Применение их.
 - *safety*
 - проверка *keepalive* пульта (гар 10 сек). Авария!
- софты
 - как в фон запускать? супервизор какой-то.
 - обучить свою модель персоной, пока что. Ездить по участку и собирать датасет ориентиров
 - режим “авария”, контроль моторов в контроллере
 - если телега должна стоять, но едет хоть одним мотором, то через 5 секунд в аварию;
 - если телега должна ехать, но все моторы стоят, то же.
 - ~~— виртуализация — ROS gazebo модель бота~~
- железо
 - старую мобилу подкинуть (GPS, IMU, LTE, камера?)
 - АЦП ADS1115 на апельсина кинуть через мою extension board
 - датчики тока на него
 - камеры надо втыкать в USB3 хаб;
 - хаб надо питать от 12B DC/DC.
 - хорошо бы иметь стоп кроме рубильника на боте. Может giro, может BT, может радио-клава, LoRa! - <https://circuitdigest.com/microcontroller-projects/raspberry-pi-with-lora-peer-to-peer-communication-with-arduino>
 - какой-то экранчик бы кинуть на третий этаж (заодно второй закрыть фанеркой). Может шпильки нарастить придётся. Заодно камеры поднять бы на третий этаж - совсем низко они
 - чёт текстовое и шустрое, типа современных ncurses;
 - моща моторов, по 4 углам “X”
 - тревоги по 4 углам “+”
 - тревоги спереди / сзади (YOLO, RGBD)
 - купить
 - нужны ещё три широко-камеры.
 - заказать новую BMS - по BT забирать статистику от BMS
- навигация (рассуждения)
 - У меня есть несколько задумок по навигации. Пока что набираю пул идей, формирую тестовые сценарии для TDD / BDD этапа потока разработки.
 - бот планирую использовать в знакомом ему окружении (на моём

- гектарном участке);
 - в четырёх камерах, охватывающих 360° вокруг, бот будет видеть привычные ему ориентиры. Если таких ориентиров 2+, то с неплохой точностью можно сориентироваться как по азимуту, так и триангуляцией найти свои координаты;
 - добавлю к этому данные GPS и IMU, считанные со старой андроид мобилы, привязанной к боту;
 - добавлю одометрию и уже получу понимание, где сейчас бот. Имея такую ОС можно строить маршруты и пытаться выполнить их, постоянно корректируя движение.
- Да, я знаю что похожая задача уже решена в ROS2 немного иначе. Это я буду всё проверять и тестировать. К сожалению, пока что я только "слышал звон" и не готов предметно обсуждать задачу.
- Но, в основе моего видения навигации бота модель, похожая на заложенную в мою голову. Я тоже вижу дом и понимаю, что моя цель в той стороне. Вижу сарай и понимаю, что я где-то рядом и т.п. Используя два глаза (RGBD у бота) я ощущаю объём. Должно всё получиться!
- Лидар
 - купить xiaomi на али
 - https://wiki.ros.org/xv_11_laser_driver
 - <https://github.com/kaiaai/kaiaai/>
 - https://www.youtube.com/watch?v=STbCVhdgLsw&ab_channel=Maker%60sPet
 - https://www.youtube.com/watch?v=mbxZAIjE8A&ab_channel=EngineerM - процесс
 - https://www.youtube.com/watch?v=WMxK5FvIEKs&ab_channel=EngineerM - результат
 - https://github.com/engineerm-jp/vacuum_robot_lidar - код
- RealSense (+SLAM)
 - <https://github.com/IntelRealSense/librealsense/blob/master/doc/installation.md>
 - https://github.com/IntelRealSense/librealsense/blob/master/doc/distribution_linux.md
 - <https://github.com/IntelRealSense/realsense-ros>
 - https://intel.github.io/robot_devkit_doc/pages/rs_slam.html
- тележка
 - мощные самодельные покрыва, то есть "цепи" напечатанные
- виртуализация
 - <https://www.youtube.com/watch?v=mDHActiZi50>
 - <https://mymodelrobot.appspot.com/6246284478382080> - онлайн визуализация URDF
- управление жестами руки
 - <https://github.com/midnightcodex/HandTrackingModule>
- Android
 - GitHub - HemaZ/ros2_android: ROS2 example packages to use your android device sensors
https://github.com/HemaZ/ros2_android
 - GitHub - sloretz/sensors_for_ros: An Android app that publishes sensor and camera data on ROS 2 topics
https://github.com/sloretz/sensors_for_ros

- android_sensors_driver - ROS Wiki
http://wiki.ros.org/android_sensors_driver
- рука (манипулятор)
 - <https://youtu.be/p5V3o9pZLVE>
 - <http://thor.angel-lm.com/documentation/>
 - <https://www.orangeapps.de/>

История

Переделать вторую STM32 на датчики

[25 марта - 12 апреля 2025]

  Почему я выбрал Black Pill вместо Blue | STM32 + FreeRTOS для робота 🤖 #diy #...

— ~~определить состав датчиков сегодня и на завтра~~

- одна i2c шина:
 - магнитометр
 - imu
 - bme280
 - 16 канальный PWM/Servo PCA9685

— ~~8 gpio на сонары~~

— ~~(жду али) 4ch мосфеты~~

— ~~(жду али) разъемы (4x4 сонары + 5x3 i2c + 6[4ch+питание]x1 реле)~~

— ~~потянет ли мелкая пилуля или лучше сразу взять чего больше?~~

~~Мне, всё равно, проще паять, а не перепаяивать.~~

- рисовать схему
- firmware + ros2 node
 - освоить BlackPill
 - освоить FreeRTOS
 - <https://youtu.be/6LoRJO7bZal>
 - <https://chat.deepseek.com/a/chat/s/c5b24f03-82f2-4ef4-b249-c6a1c4ca8ca6>
 - <https://chatgpt.com/c/67e25d55-0fd4-800c-adc3-d25845a9e44f>
 - <https://chatgpt.com/c/67e2638d-73b8-800c-9191-7e2957f66982>
 - запустить пару датчиков на соплях
- паять на плату, как разъемы придут
- софт опять
 - запустить все датчики на напаянной плате
- проложить в коробах крутые шнуры

Переход на x64 + nvidia комп

  Апгрейд ROS2 робота: Intel + NVIDIA 3060 вместо ARM + NPU — хватит ли для И...

[⚡ Не все STM32 одинаково полезны... Я проверил! !\[\]\(b39c89771cd6fb2128a8c57aa7d97f9a_img.jpg\) #bluepill #matrix #stm32 #ros2 #diy - YouTube](#)

[18-?? марта 2025]

FPV тест на улице. Исправления до и после

   Весна 2025 и робот  #ros2 #robot #весна

[5-11 марта 2025]

Дисторсию правил генетическими алгоритмами.

  Генетические алгоритмы vs Fisheye  Тюнинг параметров  #fisheye #python #o...

[27 февраля - 2 марта 2025]

Отлаживал поведенческие алгоритмы

  ROS2 робот: Мой каркас поведенческих алгоритмов на простом примере  #diy ...

[23 - 27 марта 2025]

Обучал YOLO

 YOLO на максимум!  Датасет за полчаса с Grok 3 или ChatGPT  #yolo #grok #ch...

[20 - 22 февраля 2025]

Выбор нового проца / компа

[18-20 фев 2025]

Думал-размышлял

Блок камер

[27 янв - 15 фев 20



Весной я купил три камеры.

- нарисовать и отпечатать кронштейн, тестово приладить камеры 😊
- установить туда 4 камеры (три вебки и одну RGBD), спустить вниз шнуры;
- провести вниз USB, закрепить парой стяжек;
- воткнуть в хаб;
- протестировать стабильное чтение резолв и safety со всех RGB потоков
- протестировать персону в перехлесте всех камер.

Подключить периферию

[5-20 января 2025]

I2C девайсы

- сейчас у меня магнитометр на второй I2C шине апельсина. Это хорошо! Пусть так и остаётся;
- на ту же шину, пожалуй, добавлю один BME280;
- на неё же, пожалуй, добавлю один ADS1115:
 - два его входа я повешу на датчики тока колёс;
 - два других входа ... пока свободны, если я запущу замер тока через BMS.
- нужна ROS нода сборщик данных датчиков.
Конфиг используемых датчиков через параметры ноды в launch.py.

Сонары HC-SR04

Выглядит не сложно по коммутации. Надо потестировать их одновременную работу. Можно ли trig слать сразу четырём датчикам (расположенных крУгом)? И тогда логичный вопрос, нафига 4 выхода.

<https://tutorials-raspberrypi.com/raspberry-pi-ultrasonic-sensor-hc-sr04/>

Напилить ROS-ноду "сонирования" 😊

Конфиг портов через параметры ноды в launch.py.

BMS

Долго провозившись, перенёс на следующий этап (см. ниже).

Реле

Пока не используется и подключать не буду. Но keep in mind забронировать ноги под неё. Тут всё просто - кол-во GPIO по кол-ву каналов.

Напилить простую ноду для установки реле и возврата статуса (чтение из GPIO). Пока без реле, просто GPIO.

Конфиг портов через параметры ноды в launch.py.

Видео о том

"Настройка периферии робота: ROS2 и AI-кодинг   

I2C пилюль задействовать

[25 декабря 2024]

Попробовать I2C связь с апельсином. Иначе придётся на борт ставить USB-UART. Что, как минимум, мне кажется неверным. Одно дело программу заливать через переходник. Но постоянно с ним существовать перебор.

LLM пишет (проверить)

- wiring:
 - STM32 PB7 (SDA) → Orange Pi SDA
 - STM32 PB6 (SCL) → Orange Pi SCL
 - GND to GND
 - Add 4.7kΩ pull-up resistors between SDA/SCL and 3.3V.

Левая / правая пилюля будут на разных адресах. Ещё там будут BME280 и АЦП.

Возможно, на другой I2C шине. Их в апельсине много! Включаются по `orangepi-config`

РЕЗЮМЕ (ОТМЕНА): Всё настроил и работает. И понял я, что в апельсине три свободных UART на разъёме расширения. Что смысла в I2C именно для колёсных пилюль нету.

Просто повешу их на UART1 и UART2 и разгоню до любой разумной скорости

ПИД регуляторное управление колёсами blue pill с энкодером

[20-22 декабря 2024]

Энкодер работает. Одна платка готова. Надо подключить и проверить.

- UART разъём собрать в единое целое и проверить blink;
- разъём мотора - отметить ключ;
- собрать разъём энкодера;
- i2c пока забить. Проверяю по UART.

PID библиотека знакома. LLM написал `firmware`. Остаётся её отдебужить через Ardu Serial Monitor. Протестировать так и сяк. Подобрать базовые значения регулятора.

BluePill на макетку

[19 декабря 2024]

Платка простая. В каждой "ардуине" четыре разъёма:

- мотор. Тот же разъём, что и в старой арду. Вход энкодера не используется;
- энкодер:
 - питание;
 - A;
 - B.
 - земля;
- I2C шина и питание:
 - питание;
 - SCL;
 - SDA;
 - земля.
- UART и питание:
 - питание;
 - RX;
 - TX;
 - земля.

Разбудить робота!

[18 декабря 2024]

Страшновато, но:

- убрать хлам с робота - почистить сверху лишнее 😊
- вывесить колёса. Подключить их;
- проверить питание, врубить всё
- монитор к апельсину, новый WiFi и SSH в апельсин;
- врубить колёса и клавиатурой двинуть вперёд-назад с ардуин.

Первым делом нужно задействовать уже установленные энкодеры

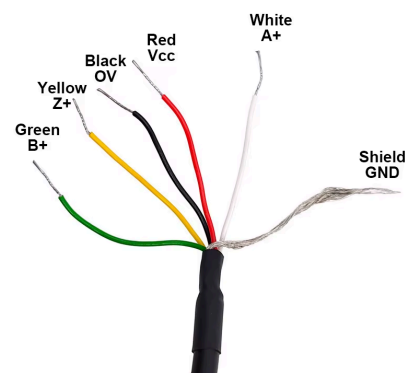
[15 декабря 2024]

По тысяче тиков в каждом; редуктор 1:4; да ещё и две фазы энкодеров. В итоге аж 8 тысяч тиков на круг колеса. Важный плюс квадратичного энкодера - теперь ясно фактическое направление движения колеса.

Десяток килогерц тиков энкодера требует stm32 с его аппаратной поддержкой энкодеров. Уже разобрался с blue pill, уже запустил левый энкодер.

- Этап "Сонары и АЦП на апельсин" [04/04/2024 - 20/01/2025]
 - Сценарий:
 - сонары занимают много арду времени. А им нужно управлять моторы. Если сонары и останутся, то их стоит перенести на апельсин.
 - Железо
 - сонары подключить к апельсиновым GPIO через резисторы - <https://clover.coex.tech/ru/sonar.html>
 - I2C АЦП подключить к апельсину
 - вообще уже просится плата расширения... Хотя бы на монтажке
 - Софты
 - firmware
 - уход от сонаров на арду - повышаю FPS, делаю его стабильным. Соответственно, корректирую протокол выдачи;
 - уход от АЦП замеров.
 - Ardu
 - убрать публикацию сонаров и температур;
 - убрать публикацию и сохранение тока;
 - Sonar
 - новая нода, напрямки на апельсине читает и публикует все N сонаров.
 - ADC
 - новая нода, напрямки на апельсине читает и публикует все N АЦП.
- Этап: "энкодеры на колёса" [~01/04/2024 - 24/12/2024]

- Сценарий:
 - не хватает разрешения встроенного в колесо энкодера (на основе 1..3 датчиков Холла);
 - в интернетах подсмотрел метод крепления внешнего энкодера;
 - там же подсказали, что STM32 железно поддерживает энкодеры!
- Железо:
 - куплено два энкодера;
 - отпечатаны две пары шестерней;
 - большие шестерни наклеены на колёса;
 - малые шестерни торчат на энкодерах;
 - отпечатан чуть пружинящий крепёж энкодера;
 - куплены STM32 синие пилули - будут считать тики и ПИД-регулировать колёса.
 - энкодеры
 - ~~— рисовать и печатать главную шестерню~~
 - рисовать и печатать малую шестерню
 - ~~— клеить и центровать большую шестерню на оба колеса~~
 - ~~— рисовать, печатать и крепить пружинный крепёж энкодера~~
 - **stm32 blink**
 - **энкодер к stm32 и писать новый софт**
 - **firmware**
 - **ros module**
- Софты
 - arduino -> stm32 переход. Там нативная поддержка энкодеров плюс либа ПИД-регулирования колеса. Протокол UART общения, пока тот же, что был;
 - на ROS стороне, стало быть, изменений минимум. Константы резко увеличатся. Так как разрешение энкодера каждого колеса поднялось в $4000 / 15 =$ в 266 раз!
- Этап "Плавный ход" [15/12/2024 - 24/12/2024]
 - Сценарий:
 - проблема с малой плотностью тиков энкодера, построенного на одном датчике холла. Сначала было думал повысить плотность за счёт использования трёх датчиков холла. Но потом заказал внешний 1000-тик энкодер;
 - чтобы повысить частоту коррекции, арду будет заниматься только ПИД-регулированием. Протокол будет изменён;
 - atmega не тянет такую плотность тиков. Но в stm32, оказывается, есть аппаратная поддержка энкодера! Так что по одной BluePill на мотор и софт ПИД-регулятора.
 - Железо
 - Заказал и получил пару энкодеров



- Напечатал пару пар шестерён и кронштейнов:



- Купил пару BluePill



- **монтажка, туда пилюлю и три разъёма**
 - энкодер;
 - контроллер мотора;
 - I2C на будущее (пока отладка по USB-UART).
- Софты
 - попросил АЙ написать *firmware* и переделать *ros-module*.
Отладку начал с *blink* 😊
Протестировал и отладил отдельный тест энкодера, написанный АЙ.
Запустил полноценную *firmware* с контролем из терминала.
 - протокол общения ардуины с ROS::Ardu упростил
 - $S = stop$
 - $M <+-NNN>$ = скорость движения вперёд / назад
 - *firmware*
 - уход от двух моторов на одной арду *bluepill*. Аппаратная поддержка энкодера.
 - Ardu
 - уход от PID регулирования на стороне ROS. Теперь этим занимается ардуина;
 - смена протокола общения с ардуиной на ПИД-регулирование.
- Этап "Датчики старой мобилы" [06/04/2024 - ??/??/2024]
 - Сценарий:
 - У меня есть Xiaomi 4 ненужный. А у него есть очень нужные мне датчики. Хочу все их видеть в ROS топиках!
 - GPS;
 - IMU;
 - ???
 - Ещё хочу получить LTE связь через мобилу.
 - Железо

- Мобила, подключенная MicroUSB (разъем залит термоклеем!)
- Софты
 - на мобиле:
 - перебрать разные софтины из списка. Возможно, найти какие-то ещё. Как я понимаю, основная проблема - поддержка ROS2;
 - ROS:
 - пока не планирую никакого использования. Этап завершается данными, регулярно публикуемыми в топики
- Этап: "веб сервер и пульт". Часть 2, "функционал необходимый" [25/03/2024 - 31/03/2024]
 - Сценарий:
 - Есть каркас пульта. Для успешной настройки бота нужно развить функционал:
 - показ скоростей и мощностей моторов;
 - показ режима (mode) и деталей контроллера:
 - режим "останов перед препятствием";
 - режим "двигаться вперед с поворотом к персоне"
 - action:
 - нет персоны;
 - персона есть;
 - потерял персону (нет 3+ сек).
 - азимут персоны;
 - скорость поворота (+-1);
 - флаг движения вперед.
 - установка параметров:
 - ardu:
 - MotorPID: p, i, d.
- Софты:
 - server
 - подписка на MotorPID и MotorMove:
 - добавление в /status MotorPID и MotorMove.
 - подписка на статус контроллера:
 - добавление в /status полученных данных;
 - получение POST /control параметров и (? как лучше ?) передача их в ardu и controller.
 - index.html
 - отрисовка показаний моторов (мощность и скорость);
 - вывод текстом статуса контроллера (режим, параметры);
 - форма установки параметров.
 - ardu;
 - controller
 - публикация статуса в виде custom сообщения Status;
 - vedrus_interfaces
 - новый StatusItem
 - string название
 - string value
 - новый Status

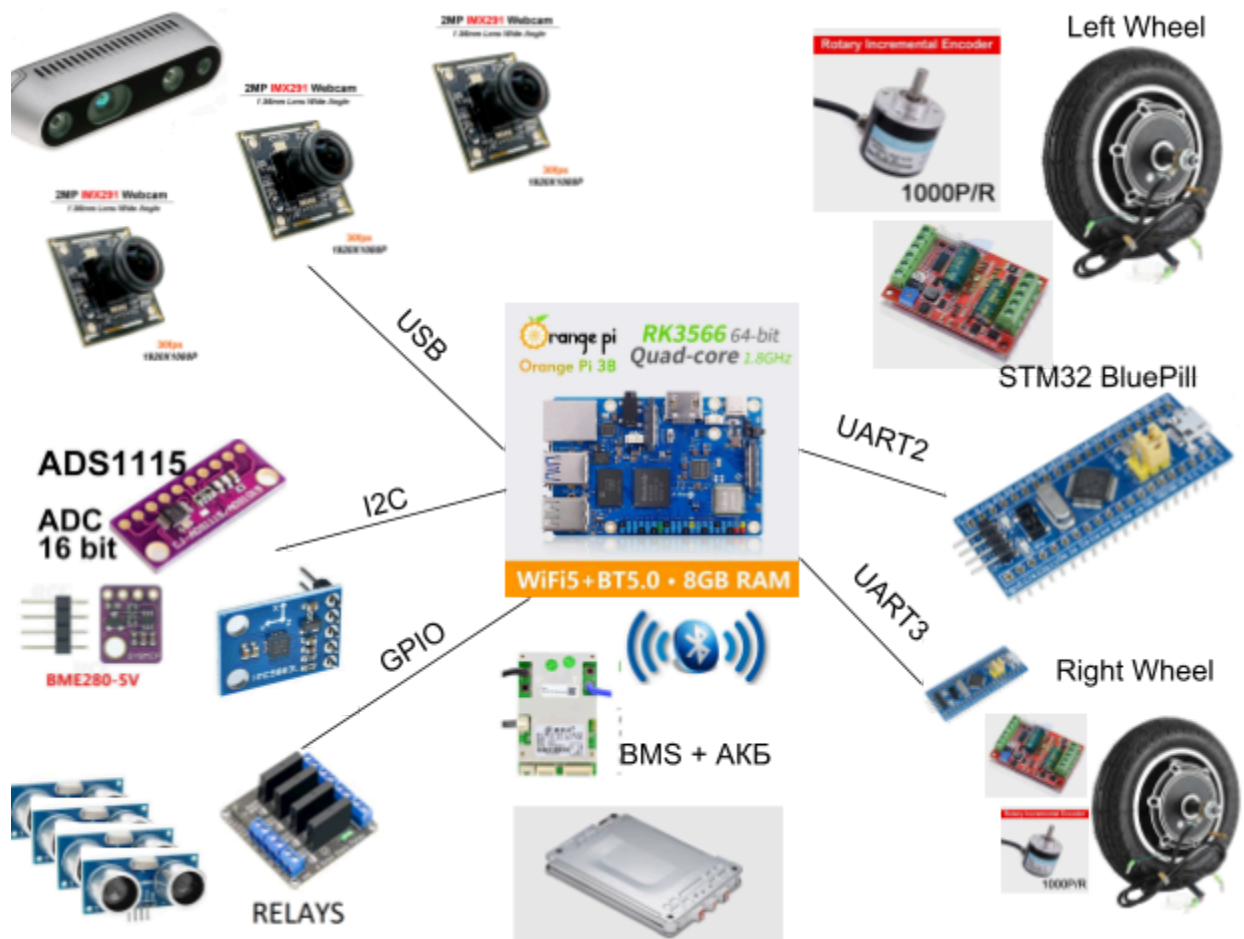
- header
 - frame_id = controller
 - текущий режим в одном из StatusItem[name=mode]
 - StatusItem[]
- ardu
 - подписка на изменение параметров от server. Применение их.
- Этап: “веб сервер и пульт”. Часть 1, “каркас” [24/03/2024 - 24/03/2024]
 - Сценарий:
 - Три колеса запустил норм, но куда чего едет - не ясно!
 - Нужен “пульт”. Выбрал веб, как знакомый инструмент.
 - Задача этапа - собрать каркас инструмента, не наполняя его функционалом. Пусть показывает Safety, например.
 - Софты:
 - новая нода server:
 - асинхронный веб сервер на aiohttp
 - js код для отрисовки safety
- Этап: “трёхколёска!” [23/03/2024 - 23/03/2024]
 - Сценарий:
 - 4 колеса не едут. Надо два и одно доПоворотное
 - Железо:
 - сварка, заварка, сверловка, рихтовка!
 - Как делать:
 - какие колёса стоят ровнее? Там перед;
 - поставить туда лучшие колёса (не левое-переднее!!);
 - снять задние колёса;
 - снять раму;
 - доварить площадку под ногу;
 - насверлить площадку, прикрутить ногу;
 - RGBD камеру вперёд;
 - тестировать этап.
- Этап: “бот, привязанный к персоне” (работает только с одной персоной. Специально нет защиты от N персон) [14/03/2024 - 14/03/2024]
 - Сценарий:
 - по RGBD находит расстояние до персоны;
 - Если расстояние больше метра, едем вперёд, не забыв подворачивать через ПИД-регулятор режиме “слежение за персоной” с доп. флагом “с движением вперёд”;
 - если расстояние 50..100 см, только поворот, флаг “движение вперёд” режима “слежение за персоной” отключен;
 - слежу за safety, как обычно. Стоп по аларму (<= 50 см до препятствия).
 - Софты
 - контроллер:
 - новая задача “safety_follow_to_person”;
 - всё как в прошлом этапе плюс...
 - режимы:
 - добавление флага “с движением вперёд” в режим

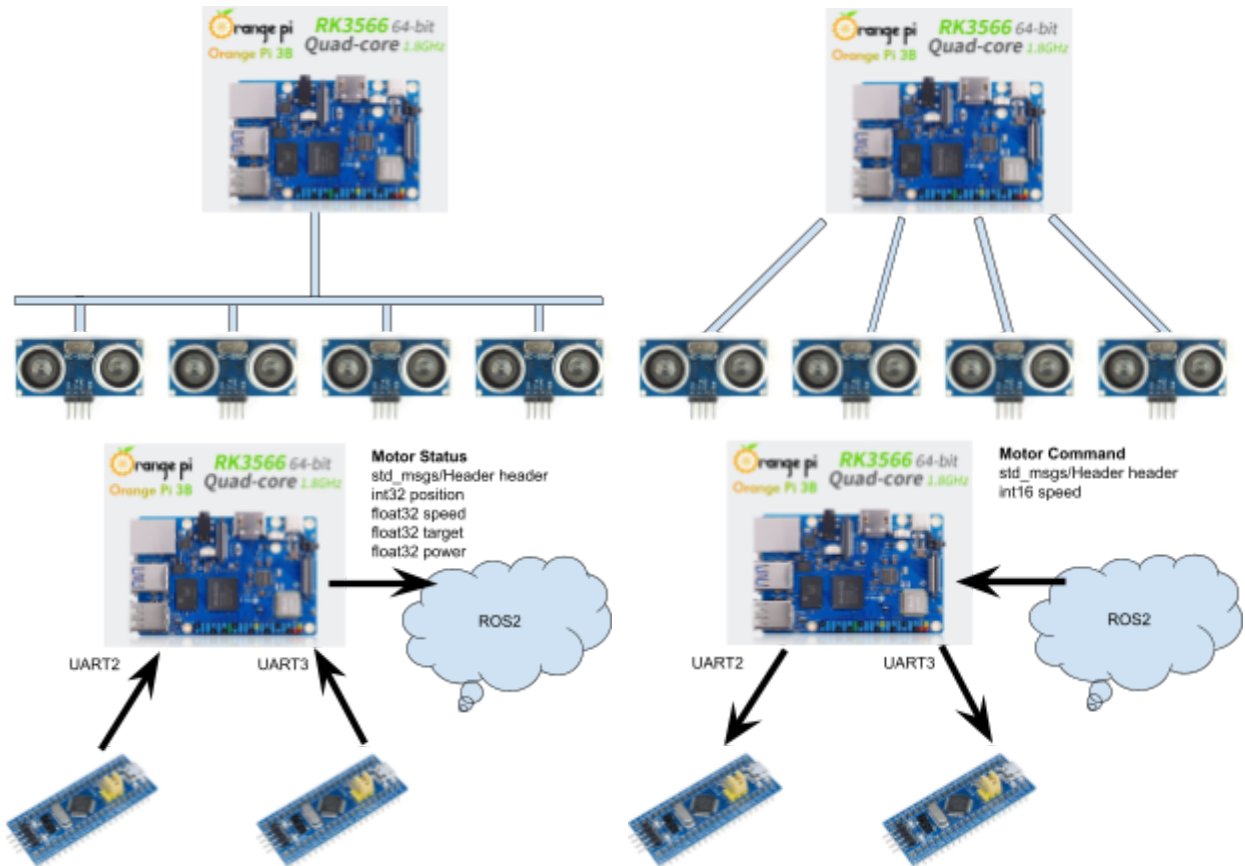
- “слежение за персоной”;
 - в этом режиме порой активируется преследование персоны по RGBD:
 - поиск расстояния до персоны;
 - флаг “движение вперёд” включается на расстоянии $> 1\text{м}$ до персоны;
- Этап “слежение за объектом по азимуту, стоя на месте”: [13/03/2024 - 13/03/2024]
 - Сценарий:
 - анализ азимута `safety::yolo::person`. Если N раз с $M+$ вероятностью найдена персона, то с неторопливым пид-регулятором поворот платформы до азимута 0 по отношению к персоне;
 - помнить о безопасности. Останов при $< 50\text{см}$ ALARM.
 - Софты:
 - контроллер:
 - новая задача “`safety_rotate_to_person`”;
 - обработка `safety` из прошлого этапа;
 - задача не использует режим езды. Но добавляет новый режим “слежение за персоной (`rotate_to_person`)”. В этом режиме, по подписке на `safety::yolo`, ищем азимут персоны и кидаем его в $-+1$ пид регулятор поворота.
 - пид регулятор поворота живёт в таймере, что логично.
 - -1 в его `set point` означает левым моторам назад, правым вперёд на N (5, например);
 - $+1$ наоборот;
 - но, конечно же, это аналоговое управление.
- Этап “движение вперёд по прямой медленно с остановкой у препятствия”: [11/03/2024 - 12/03/2024]
 - желаемый сценарий:
 - ставлю бота на площадку перед домом;
 - он (неспеша!) едет вперёд. Подъезжает к препятствию (0.5м до него) и останавливается. Препятствие с любой стороны!
 - препятствие отходит / пропадает, бот подъезжает на 0.5м опять. И так пока не надоест (мне).
 - софты
 - ввод понятия “задача”. Это `python` функция, переключающая режимы. Сейчас задача “`safety_forward`”. Начальный режим - останов перед препятствием;
 - `launch/yolo.py` работает себе. Обрабатывает команды к моторам, выдаёт `safety`;
 - `safety`
 - 3 раза в сек слать `keepalive` сообщение `std_msg/header` в тему `/vedrus/keepalive/safety`;
 - следить за `keepalive` от контроллера. Пусть друг-друга проверяют:
 - если контроллер мёртв в течение двух секунд, слать моторам STOP;
 - дать время на раскочку (10 сек с момента старта).

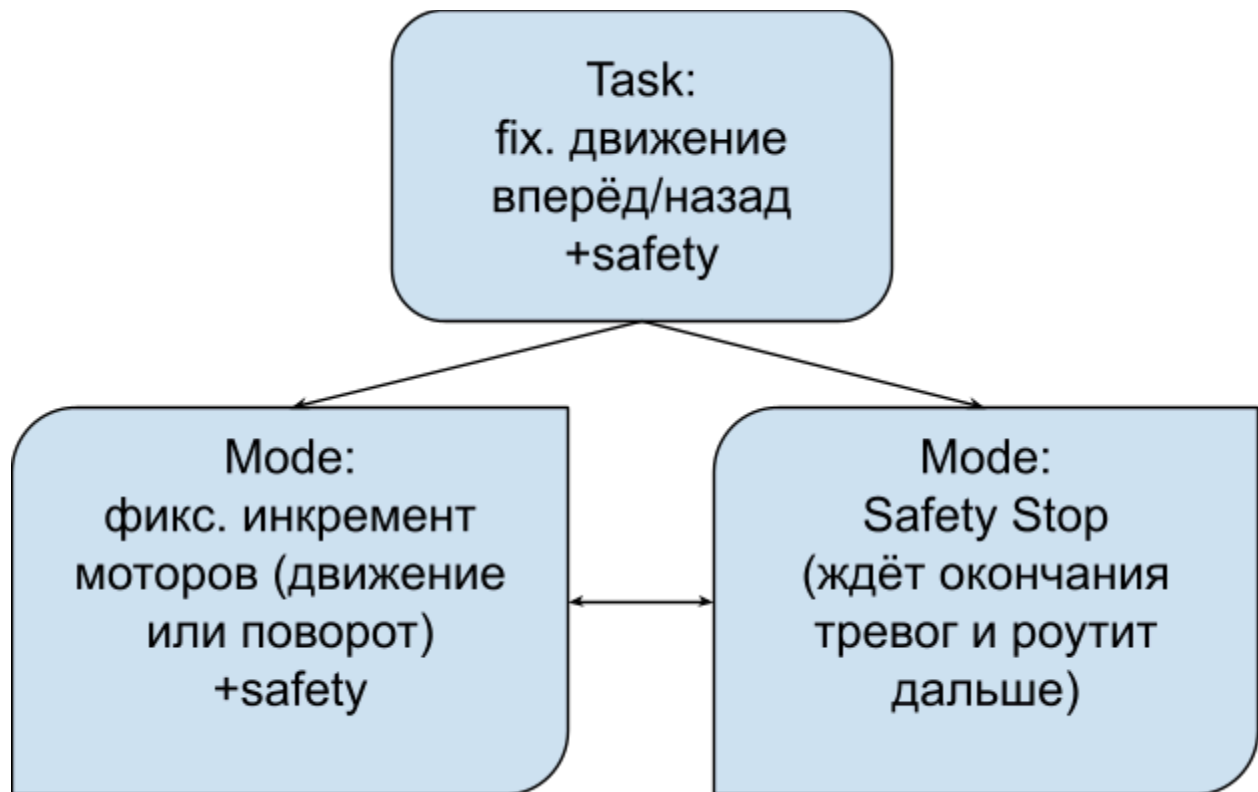
- vedrus_interface/MotorMove:
 - добавить bool флаг CRASH.
- vedrus/ardu:
 - обработка MotorMove::CRASH.
Выдать нули в мотор и больше ничего не принимать!
- контроллер:
 - 3 раза в сек слать keepalive сообщение std_msg/header в тему /vedrus/keepalive/controller;
 - следить за keepalive от safety. Пусть друг-друга проверяют:
 - если safety мёртв в течение двух секунд, слать моторам STOP;
 - дать время на раскачку (10 сек с момента старта).
 - есть разные режимы контроллера:
 - режим “останов перед препятствием” (safety_stop):
 - начальный режим задачи
 - выход из режима после двух секунд отсутствия alarm safety - переход в режим “вперёд неспеша”;
 - режим “вперёд неспеша”:
 - вход в режим: подаёт на pid моторов +5.
 - выход из режима: когда от safety получено два сообщения alarm (препятствие ближе полуметра) в течение 0.5с. Команда pid 0;
 - останов с тормозом
 - подписки:
 - Safety/keepalive
 - Safety
- Этап “набор скорости до 100 tick per second” [06/03/2024 - 09/03/2024]
 - сценарий:
 - колёса вывешены. Или я их торможу. Или бот на земле
 - даю команду “установи скорость левых/правых колёс в xxx тиков в секунду (направление - вперёд/назад)!”
 - колёса плавно раскручиваются / замедляются
 - при пробуксовке (свободном вращении) одного колеса, данные его энкодера не учитываются
 - если колесо тормозит, ток ему добавляется. При этом контролируется момент срыва и мощность резко убавляется!
 - алгоритм
 - через simple_pid впендюрить ПИД-регуляторы на все колёса (стороны?). А куда? Видимо, в Ardu::timer (5 fps)
 - антипробуксовка: если тики сильно отличаются, считаю что одно колесо крутится свободно - учитываю только минимальные тики;
 - железо
 - зафиксировать / снять BMS. Спрятать ее провода (временно, пока не пришла BMS);
 - куда-то положить батарейку мелкую, основная не активна из-за BMS
 - приклеить рубильник
 - левая сторона

- **заменить** датчик тока (пока на 20 или какой есть)
 - **разводить** питание толстым проводом
- правая сторона
 - **разводить провода** толстым проводом
 - 2 датчика тока
 - Smart BMS приделать и подключить пока к мобиле.
- Этап "безопасность" [??? - 06/03/2024]
 - сценарий:
 - бот стоит на месте или едет - не важно
 - работает нода yolo, шлёт ресолвы;
 - работают ноды камер, шлют RGB и глубину (для передней камеры);
 - работает нода safety, шлёт тревоги.
 - железо
 - сонары боковые подключить;
 - задний сонар приклеить и налево подключить (пока не пришёл лидар)
 - убрать USB хаб (пока не пришёл DC/DC на 12)
 - RGBD и RGB камеры в апельсин USB напрямую
 - софт
 - ardu-firmware:
 - -1 сонар справа (выдавать ноль)
 - добавить мощность моторов
 - добавить dir моторов
 - прошить и проверить fps обоих
 - ardu-node:
 - дописать header в заголовок сообщения msg.Motor
 - чтение из порта мощности и dir, дописать в msg.Motor
 - задний сонар публиковать в /vedrus/sonar::id=side_rear
 - дебаг изменений
 - vedrus_interfaces
 - добавить интерфейс Sonar
 - добавить интерфейс Safety
 - header
 - bool warning
 - bool alarm
 - float32 range
 - float32 azimuth
 - собрать
 - yolov8_rknn/solver
 - а не сдохнет ли от двух загруженных моделей?
 - Можно мою и общую YOLOv8 прогонять
 - азимут объекта найти по X центра, добавить в inference.
 - vedrus/safety
 - rgbd
 - подписывается на /depth/image_raw
 - медианой моем
 - находим центры больших минимумов
 - если расстояние до большого объекта в центре

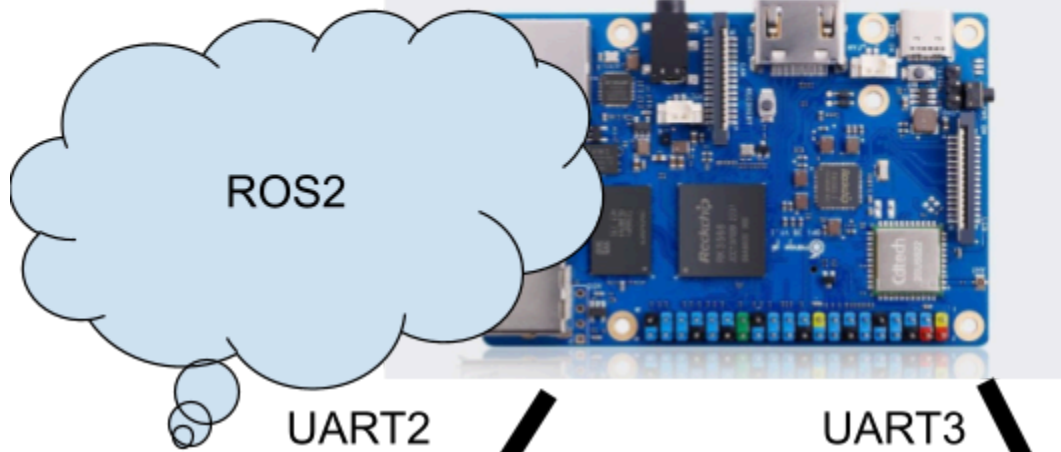
- меньше метра, шлём msg.Safety.Warning
- если расстояние до большого объекта в центре меньше полуметра, шлём msg.Safety.Alarm
- если расстояние до большого объекта не в центре меньше полуметра, шлём msg.Safety.Warning
- сонары
 - подписываемся на /vedrus//sonar
 - если расстояние до объекта меньше метра, шлём msg.Safety.Warning
 - если расстояние до объекта меньше полуметра, шлём msg.Safety.Alarm
- резолвы
 - если найден объект из списка (человек, кошка, собака, машина, вел, моц, ??), если размер объекта больше N пикселей (размер на каждый класс), прислать msg.Safety.Warning

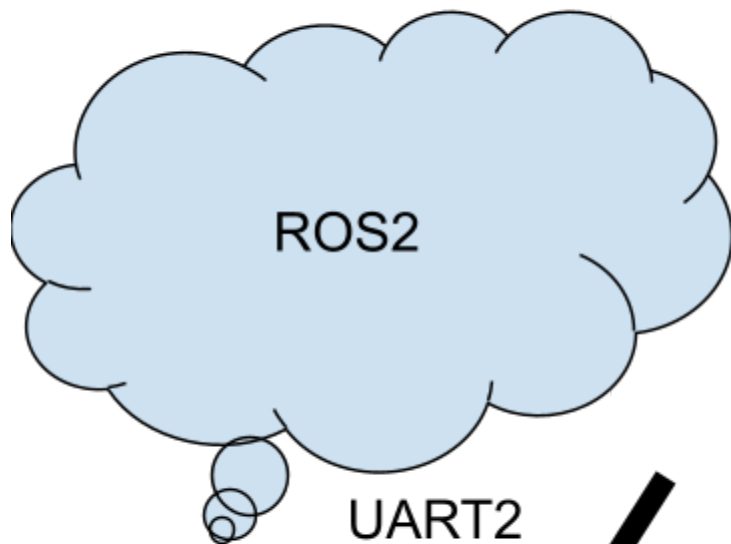






orange pi **RK3566** 64-bit
Orange Pi 3B Quad-core 1.8GHz





Rotary Incremental Encoder



1000P/R



Rotary Incremental Encoder



1000P/R