Команда “Таёжные Ёжики”

Состав команды:

Пильщиков Григорий

Цыганкова Мария

Предназначение
Задачей команды является создание робота и написание ПО для обследования лабиринта и построение карты, обнаружения жертв и доставки необходимого количества медкомплектов.

Техническое описание разработки
Шасси робота сконструированы на базе конструктора makeBlock. На роботе стоит камера, которая находит метки на стенах при помощи микрокомпьютера Raspberry Pi Zero и инфракрасные дальномеры sharp 2y0a21 f 36.

Стратегия
На данный момент наш робот ищет в лабиринте “жертв” по правилу "чередующийся руки". Далее мы планируем сделать более гибкую стратегию, в будущем мы будем строить карту лабиринта. Робот будет запоминать пройденный маршрут, и даже при замкнутых системах найдет выход из него.

Использование датчиков
В нашем роботе используется 3 инфрокрасных дальнометра sharp 2y0a21 f 36. Камера, осчитываемая Raspberry Pi Zero, которая передает данные на arduino. Также в нашем роботе есть встроенный датчик: гироскоп, который используется для поворотов робота и подъёме по рампе, помимо этого в нашем роботе имеется датчик освещённости, используемый для обнаружения цветных клеток.

Программное обеспечение.
Программируем мы нашего робота на Arduino IDE при помощи библиотеки MeAuriga.h
Также мы программируем техническое зрение на C++ с использованием библиотеки OpenCV.
Кроме того в процессе разработки ПО команда практиковалась в симуляторе Webots, где был разработан алгоритм движения робота по лабиринту и азы построения карты.

Уникальные алгоритмы
Жертвы и Hazmat signs мы обнаруживаем с помощью датчика камеры и библиотеки OpenCV 4. Для начала настраивается бинаризация для 1 из 3 цветов: красного, жёлтого и чёрного. После бинаризации определяем контуры и проверяем самые большие контуры, подходят ли они. Далее если нашлись подходящие контуры для жёлтой бинаризации, то мы определили Hazmat sign- Organic Peroxide, если же не найдены подходящие жёлтые, но есть подходящие красные, то это Flammable Gas, если же обнаружены чёрные контуры, подходящего размера, то вычисляем 3 точки прямоугольника этого контура: верхнюю срединную, среднюю срединную и нижнюю срединную, с помощью них мы и определяем какая буква и ищем в них контур на чёрной бинаризации: если верхний, средний и нижний - чёрные, то это s, если нижний и верхний - не чёрные, а средний - чёрный, то это H, а если нижний - чёрный, а средний и верхний - нет, то это U.

Для построения карты наша команда решила использовать двумерный массив из структур, в каждую из которой при посещении мы вносим данные об наличии жертвы в ней, если жертва есть, то о её типе, о типе пола (старт/чекпоинт/обычный и т.д.), о наличии стенок вокруг тайла и о количестве прохождений этого самого тайла. На этих данных и строиться алгоритм поведения нашего робота в лабиринте. На данном этапе мы не имеем никаких slam и a* алгоритмов. Наш робот, просто на основе посещений тайлов выбирает по какой руке ехать: по правой или по левой. А с помощью запоминания ячеек, в которых находятся жертвы наш робот не распознаёт их дважды

Решение проблем.
Наша команда столкнулась с проблемами.
Первая проблема была с командами из библиотеки MeAuriga.h. Мы не могли найти подходящие команды для программирования робота. Решением проблемы стало перебирание примеров из этой библиотеки, благодаря этому нам удалось найти нужные команды.
Вторая проблема заключалась в реализации бинаризации технического зрения. Решение этой проблемы стало тщательное изучение темы “бинаризация” и просмотр обучающих видео по этой теме.

Ссылки на нас
Наш Github - https://github.com/Grin2020/TE2022;
Наш youtube канал: https://youtu.be/IK50Rj7IdK4