**УДК 62-52, 001.57, 004.358, 004.942, 519.876.5**

# РАЗРАБОТКА НЕЙРОЭВОЛЮЦИОННОГО КОНТРОЛЯ МАШИНЫ С РУЛЕВЫМ УПРАВЛЕНИЕМ АККЕРМАНА В СИМУЛЯТОРЕ V-REP

Д.Ю. Ларионова (*d.larionova@innopolis.ru*)
М.А. Иванов (*m.ivanov@innopolis.ru*)
И.М. Афанасьев (*i.afanasyev@innopolis.ru*)

Университет Иннополис, Иннополис

**Аннотация**. Разработка интеллектуального контроля для транспортных систем является перспективной задачей современной мобильной робототехники. В этой статье разработан нейроэволюционный контроль машины с рулевым управлением по схеме Аккермана, работа которого продемонстрирована в симуляторе V-REP. Для моделирования эксперимента были разработаны 3D сцена, сценарий, модель машины, учитывающая кинематические особенности движения аккермановской машины. Для автоматического тестирования были созданы модули управления машины в среде Python Remote-API и контроля схода машины с трассы. Сперва в симуляции был исследован ПИД-контроль движения машины вдоль линии на основе показаний видеосенсоров, затем для контроля и стабилизации машины на трассе был разработан нейроэволюционный алгоритм NEAT с входными данными, как от видеосенсоров, так и от сенсоров расстояния. Работа алгоритма была проверена на разных трассах, показав, что нейроэволюционный контроль успешно справляется с управлением аккермановской машины, формируя оптимальные параметры контроля (углы рулевого управления и скорости) и оптимальную структуру нейросети для управления машиной.

**Ключевые слова**: нейроэволюционный контроль, рулевое управление машины, схема Аккермана, NEAT алгоритм, ПИД-контроллер, симулятор V-REP, моделирование движения

## Введение

Разработка интеллектуального контроля для транспортных систем является перспективной задачей современной мобильной робототехники, требующей междисциплинарного подхода. Анализ литературы показывает, что движение мобильных объектов часто изучается с использованием

моделей роботов в 3D симуляторах, таких как Gazebo [Афанасьев и др., 2015; Sokolov et al., 2016, Shimchik et al., 2016; Sokolov et al., 2017a], MATLAB [Лавренов и др., 2016; Magid et al., 2017], MATLAB/Simulink [Khusainov et al., 2016], Scilab [Dobriborsci et al., 2016], V-REP [Rohmer et al., 2013], применение среды ROS/RViz [Зенкевич и др., 2017; Ibragimov et al., 2017; Bokovoy et al., 2018; Filipenko et al., 2018; Лавренов и др., 2019], гоночных симуляторов, таких как Speed Dreams [Zubov et al, 2018], а также прикладного программного обеспечения с генерацией антропоморфных моделей [Gabbasov et al., 2015; Danilov et al., 2016]. Кроме того, подобные симуляторы используются для генерации карт и разработки специальных сред [Afanasyev et al., 2015; Lavrenov et al., 2017]. Движение машин с аккермановскими схемами рулевого управления исследовалось в работах [Shimchik et al., 2016; Sheikh et al., 2018; Filipenko et al., 2018; Zubov et al, 2018]. Использование нейроэволюционных алгоритмов для управления роботами представлены в исследованиях [Ahmadzadeh et al., 2015; Sokolov et al., 2017b]. В данной работе предложен и реализован в симуляторе V-REP нейроэволюционный метод управления аккермановской машины, формирующий команды контроля углов рулевого управления и скорости.

# 1 Процесс моделирования эксперимента в V-REP

## 1.1 Моделирование эксперимента с машиной в среде V-REP

В качестве среды моделирования эксперимента с аккермановской машиной использовался симулятор V-REP компании Coppelia Robotics. Из библиотеки V-REP была применена модель аккермановской машины (Рис. 1), с установленными на ней тремя видеосенсорами, направленными в пол для контроля интенсивности света и тремя датчиками близости (proximity sensors). В среде V-REP специально для эксперимента была создана гоночная трасса полигона (Рис. 1), состоящая из стен ограждения и дороги, с линией замкнутой формы посередине, для движения машины по замкнутой траектории. Треки дороги имеют разную форму, но общий стиль. Черная линия, вдоль которой движется автомобиль, окружена стенами на равном удалении, которые также используются для контроля положения машины внутри полигона (по равноудаленности от стен по датчикам приближения). Красные линии нужны для автоматического определения схода машины с дистанции (по видеосенсорам). Таким образом, был создан сценарий в среде V-REP в формате *.ttt, реализующий 3D сцену и модель аккермановской машины (с шасси, приводами, рулевым управлением и сенсорами), полностью отражающий кинематические особенности движения машины.

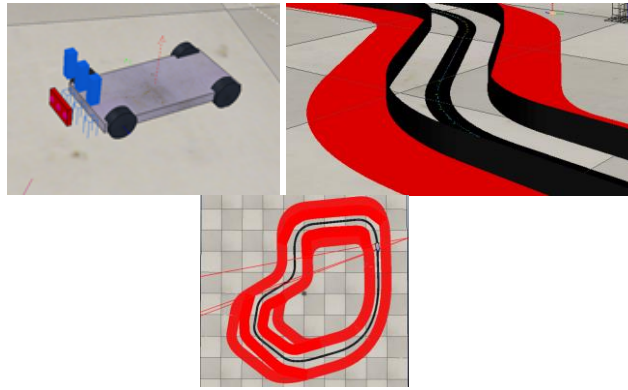

Рис 1. 3D модель машины с рулевым управлением по схеме Аккермана (слева). Смоделированная гоночная трасса полигона из стен ограждения и дороги замкнутой формы для эксперимента в симуляторе V-REP (в центре и справа)

## 1.2. Разработка алгоритма стабилизации машины с помощью ПИД-контроллера в среде V-REP

Задача заключалась в симуляции движения аккермановской машины вдоль линии (line following) со стабилизацией движения при помощи ПИД-контроллера. При симуляции машина движется по трассе полигона с постоянной скоростью (Рис. 2). Для определения положении автомобиля относительно линии движения считываются значения интенсивности света с левого и правого видеосенсоров и рассчитывается разность интенсивности света между ними. В условиях симуляции показания видеосенсора варьируются в диапазоне от 0 (черный цвет, минимальная освещенность) до 1 (белый цвет, максимальная освещенность). Если линия проходит строго между двумя сенсорами, в идеальных условиях их показания будут равны (разность интенсивности света будет равна 0). На основе разности показаний датчиков рассчитывается ПИД-коэффициенты. устанавливающие угол рулевого управления (отрицательные и положительные значения разности показаний), что в конечном счете определяет направление поворота автомобиля. Псевдокод и блок-схема алгоритма контроля движения машины показаны на Рис. 3.

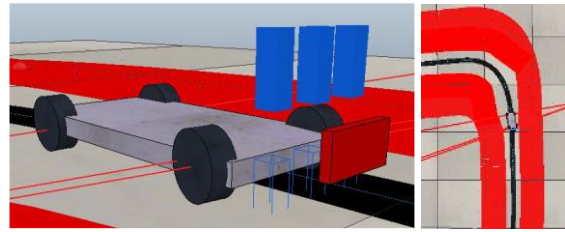

Рис 2. Симуляция эксперимента с движением машины по трассе полигона и трекингом линии сенсорами в симуляторе V-REP

```
var prev_error = 0, error, integral = 0, derivative = 0,
left_intensity, right_intensity, computed_angle
        while true
        do
                left_intensity = read_left_sensor_data ()
                right_intensity = read_right_sensor_data ()
                error = right_intensity - left_intensity
                integral = integral + error
                derivative = error - prev_error
                computed_angle =
                deg_to_radians(kp*error + ki*integral +
                kd*derivative)
                set_steering_angle(computed_angle)
        end
```

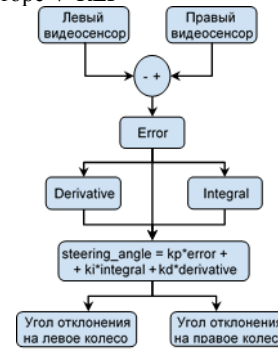


Рис 3. Псевдокод и блок-схема алгоритма контроля движения машины вдоль линии на основе показаний видеосенсоров и ПИД-контроллера

# 2 Разработка алгоритма нейроэволюционного контроля машины с аккермановским управлением в среде V-REP

## 2.1. Разработка алгоритма управления машины в V-REP

На данном этапе был разработан алгоритм, написанный в Lua-скрипте V-REP сцены. Он реализовал контроль движения машины вдоль линии на основе показаний видеосенсоров и ПИД-контроллера (см. раздел 1.2). Проект содержал всего один файл сцены V-REP для 3D модели машины с аккермановским рулевым управлением в формате ttt-файла (*.ttt). С одной стороны, этот подход удобен, поскольку запуск требует только установку V-REP и загрузку сценария (ttt-файла) с полигоном и смоделированной машиной. Так что для старта проекта не требуются дополнительные компиляторы или интерпретаторы, достаточно запустить стартовую команду в симуляторе V-REP. С другой стороны, такой подход не практичен, так как добавление новых особенностей (features) требует временных затрат для оптимизации. Поэтому потребовалась новая реализация управления аккермановской машины с последующей автоматизацией алгоритма подбора параметров. При анализе интеграции

симулятора V-REP с другими языками программирования выбор пал на Python Remote-API. В результате этого этапа, к ttt-файлу в проект V-REP добавилось еще несколько файлов (помимо API): main.py, connections.py, и AckermannCar.py. Описание назначения файлов приведено в Таблице 1. У данного программного решения оказался только один недостаток, на тот момент никак себя не проявивший: класс AckermannCar был выполнен для алгоритма следования линии, что исключало возможность для управления машиной во время симуляции другими способами.

Таблица 1. Структура проекта с контролем машины по схеме Аккермана в V-REP с управлением на основе Python Remote-API

| **Файлы проекта** | **Назначение** |
|---|---|
| main.py | Стартовый скрипт с параметрами запуска проекта |
| connections.py | Скрипт с методами установления и закрытия соединения с V-REP по API, загрузкой сцены и запуска симуляции |
| AckermannCar.py | Скрипт с описанием одноименного класса, конструктор которого принимает 4 параметра (помимо ID соединения для вызова функций V-REP): скорость автомобиля и 3 коэффициента ПИД-контроллера. Метод run_car() содержит алгоритм, описанный в псевдокоде на Рис. 3. |

Ручной процесс подбора параметров неэффективен и необъективен, поскольку может упустить из виду лучшие варианты. Поэтому появилась необходимость автоматизации тестирования с различными стартовыми параметрами. На этом этапе в проект был добавлен файл TestDataGenerator.py, который инициализировался массивом с возможными диапазонами и шагом изменения генерируемых параметров. Так, при каждом вызове get_params() генерировалось уникальное сочетание параметров, которые записывались в файл *.csv с соответствующим временем симуляции. Однако, как определить, что машина все еще следует линии? Для этого было принято решение обвести существующую трассу с обеих сторон на некотором расстоянии красными линиями, пересечение с которыми означало бы сход автомобиля с дистанции (Рис. 1). Для регистрации факта схода с трассы использовался средний видеосенсор на машине для распознавания изменения цветов (Рис. 2). Это решение оказалось удачным и использовалось для экспериментов и набора статистики прохождения трассы машиной.

## 2.2. Разработка нейроэволюционного алгоритма для управления машиной в симуляторе V-REP

Хотя ПИД-контроллер надежно контролирует движение машины, целью исследования является разработка нейроэволюционного алгоритма стабилизации машины на трассе. Чтобы использовать для управления автомобилем нейросеть требуется её обучить (т.е. подобрать коэффициенты), что не представляет проблему, если известна топология сети. Однако, составление архитектуры сети является сложной задачей. Поэтому была применена библиотека эволюционных алгоритмов NEAT (NeuroEvolution of Augmenting Topologies), помогающая подобрать топологию нейросети [Stanley et al, 2002]. Библиотека NEAT совместима с Python Remote-API, что важно для нашей симуляции в V-REP.

Принцип работы алгоритма следующий (Рис. 4). На вход подаются данные с трех сенсоров одного вида (видео или расстояния), на выходе получаем угол рулевого управления и скорость машины. Чтобы заменить в симуляции ПИД-контроллер на нейроэволюционное управление понадобился рефакторинг проекта. Теперь в классе AckermannCar остались только методы чтения данных сенсоров и изменения скорости и угла рулевого управления. Алгоритм следования линии был вынесен в отдельный класс-контроллер, что позволило быстро менять алгоритм управления машины (ПИД-контроллер или нейросеть) и модуль исследуемой задачи (следование линии, автономная навигация или др.).

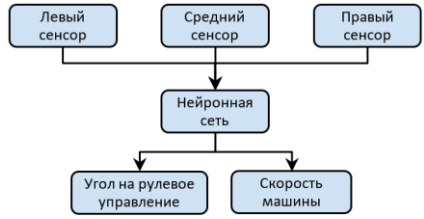


Рис 4. Симуляция эксперимента с движением машины по трассе полигона и трекингом линии сенсорами в симуляторе V-REP

# 3 Результаты работы алгоритма нейроэволюционного контроля машины в симуляторе V-REP

Сперва алгоритм тестировался на сцене ackerman_car.ttt. При одних и тех же начальных настройках геномы-долгожители (достигшие лимита по времени симуляции) появились уже на ранних поколениях при использовании видеосенсора (vision_sensor). Так, первый такой геном-долгожитель появился уже во 2 поколении (Рис. 5), тогда как у сенсора близкого расстояния (proximity_sensor) - только в 39 поколении (Рис. 6).

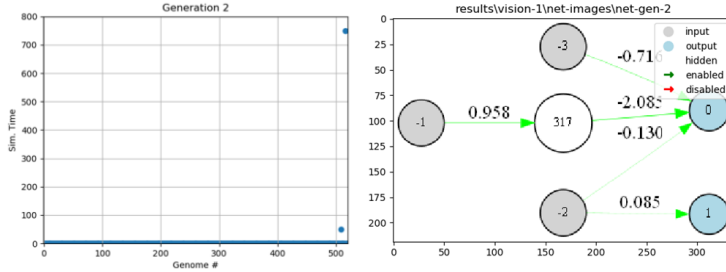


Рис. 5. Статистика генерации геномов в поколении (слева) и структура нейросети (справа) при использовании нейроэволюционного алгоритма NEAT для контроля машины в симуляторе V-REP для входных данных от видеосенсора

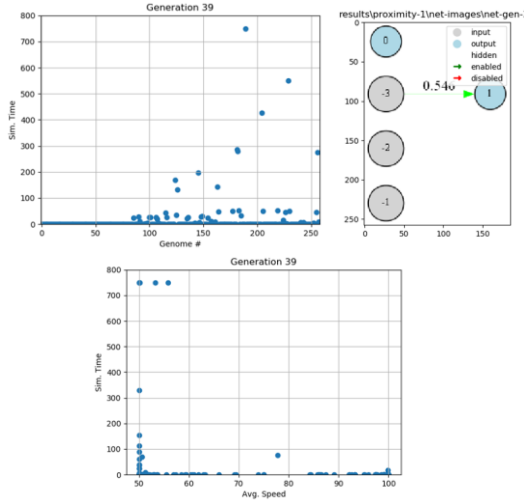


Рис. 6. Статистика генерации геномов в поколении (слева), структура нейросети (в середине) и распределение средних скоростей машины (справа) при применении нейроэволюционного алгоритма NEAT для контроля машины в среде V-REP для входных данных от сенсора расстояния

Далее поколения выбирались случайным образом из уже получившихся удачных геномов. Потом для видеосенсора на 24 поколении, а у сенсора расстояния на 42 поколении изменили сцену в симуляторе V-REP одну на другую, с более сложной трассой, записанной в файл «ackerman_car_complicated.ttt». Рассмотрим изменения результатов для видеосенсора. Сперва статистика генерации геномов в поколении испортилась (см. переход от 23-ого к 24-му поколению на Рис. 7, слева и в

середине), что не удивительно, поскольку существовавшая на тот момент нейросеть не была готова к более крутым поворотам трассы. Однако, к 39 поколению ситуация стабилизировалась и количество генов-долгожителей снова выросло (см. Рис. 7, справа). При этом структура самой сети заметно упростилась: как видно из Рис. 8, теперь для управления используется только один сенсор.

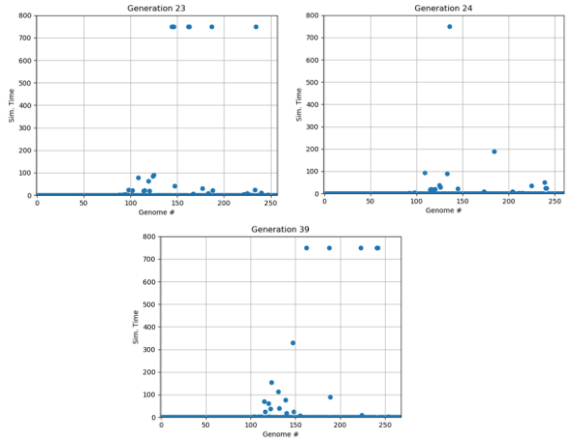


Рис. 7. Статистика генерации геномов в поколениях: 23 (слева), 24 (в середине) и 39 (справа) при использовании нейроэволюционного алгоритма NEAT для контроля машины в симуляторе V-REP для входных данных от видеосенсора

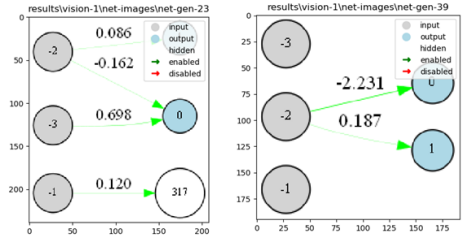


Рис. 8. Структура нейросети при генерации геномов в поколениях: 23 (слева) и 39 (справа) при использовании нейроэволюционного алгоритма NEAT для контроля машины в симуляторе V-REP для входных данных от видеосенсора

Теперь рассмотрим данные для сенсора близкого расстояния. Замена сцены на более сложную была проведена в 42 поколении, и также статистика генерации геномов испортилась при переходе от 41-ого к 42-му поколению (Рис. 9). Однако, к 56-му поколению ситуация улучшилась и генов-долгожителей стало больше (Рис. 9, справа) и структура сети стала сложнее (Рис. 10). При этом распределение средних скоростей машины для

геномов разных поколений не сильно изменилось (Рис. 6, справа и Рис. 10, справа), показывая, что более устойчивое движение машины для генов-долгожителей происходит при скорости ~50% от максимальной, что соответствует медленному движению по трассе.

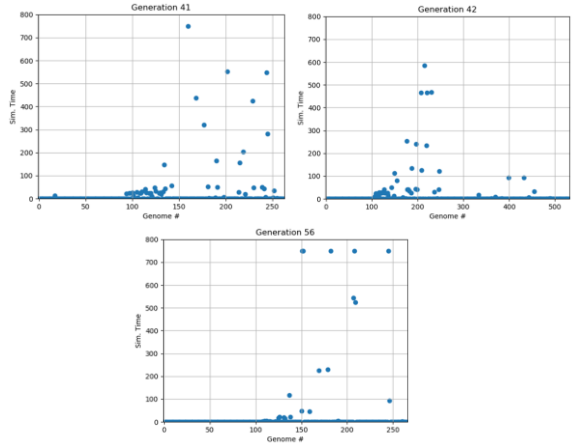


Рис. 9. Статистика генерации геномов в поколениях: 41 (слева), 42 (в середине) и 56 (справа) при контроле машины на базе NEAT по сенсору расстояния в V-REP

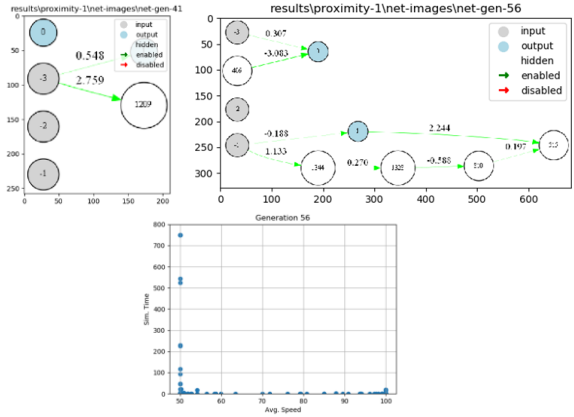


Рис. 10. Структура нейросети при генерации геномов в поколениях: 41 (слева) и 56 (в центре), а также распределение скоростей машины для 56-го поколения (справа) при контроле машины на базе NEAT по сенсору расстояния в V-REP

## Заключение

В статье предложен нейроэволюционный метод контроля машины с рулевым управлением по схеме Аккермана. Для моделирования эксперимента в симуляторе V-REP был создан сценарий с трехмерной сценой и моделью машины (с соответствующими шасси, приводами, рулевым управлением и сенсорами), движение которой полностью отражает кинематические особенности аккермановской машины. Для автоматического тестирования были созданы программные модули управления машины в среде Python Remote-API и контроля схода машины с трассы. В начале моделирования, для отработки поведения машины в симуляции, использовался ПИД-контроллер, стабилизирующий движение машины вдоль линии с помощью видеосенсоров. Затем для контроля и стабилизации машины на трассе был применен нейроэволюционный алгоритм NEAT с входными данными двух типов: от видеосенсоров и от сенсоров расстояния. Нейроэволюционный контроль машины в симуляторе V-REP был исследован сперва на одной трассе, а потом валидированы на другой, показав, что нейроэволюционный подход успешно управляет аккермановской машиной, формируя через несколько десятков поколений геномы-долгожители с оптимальными параметрами контроля (как по углам рулевого управления, так и по скоростям машины) и оптимальную структуру нейросети для управления машиной.

## Список литературы

**[Афанасьев и др., 2015]** Афанасьев и др. Навигация гетерогенной группы роботов (БПЛА и БНР) через лабиринт в 3D симуляторе Gazebo методом вероятностной дорожной карты. Сб. тр. БТС-ИИ, 18-25, 2015.

**[Зенкевич и др., 2017]** С.Л. Зенкевич, Ч. Хуа, Х. Цзяньвень. Движение группы мобильных роботов в строю типа “конвой” - теория, моделирование и эксперимент. Сб. трудов БТС-ИИ, 136-147, 2017.

**[Лавренов и др., 2016]** Лавренов Р. О., Афанасьев И.М, Магид Е.А. Планирование маршрута для беспилотного наземного робота с учетом множества критериев оптимизации. Сб. трудов БТС-ИИ, 10-20, 2016.

**[Лавренов и др., 2019]** Р.О. Лавренов, Е.А. Магид, Ф. Мацуно, и др. Разработка и имплементация сплайн-алгоритма планирования пути в среде ROS/Gazebo. Труды СПИИРАН, 18(1), 57-84, 2019.

**[Afanasyev et al., 2015]** I. Afanasyev, A. Sagitov, and E. Magid, ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment. In ACIVS. Springer, 273-283, 2015.

**[Ahmadzadeh et al., 2015]** H. Ahmadzadeh, E. Masehian. Modular robotic systems: Methods and algorithms for abstraction, planning, control, and synchronization. Artificial Intelligence, 223, 27-64, 2015.

**[Bokovoy et al., 2018]** A. Bokovoy, M. Fomin, K. Yakovlev. Implementation of the Pathfinding System for Autonomous Navigation of Mobile Ground Robot. In ITTMM-WSS, 72-78, 2018.

**[Danilov et al., 2016]** I. Danilov, B. Gabbasov, I. Afanasyev, and E. Magid. ZMP Trajectory from Human Body Locomotion Dynamics Evaluated by Kinect-based Motion Capture System. In VISAPP, 162-168, 2016.

**[Dobriborsci et al., 2016]** D. Dobriborsci, A. Kapitonov, N. Nikolaev. The basics of the identification, localization and navigation for mobile robots. In Int. Conf. on Information & Digital Technologies (IDT), 100-105, 2017.

**[Filipenko et al., 2018]** Filipenko M., Afanasyev I. Comparison of various slam systems for mobile robot in an indoor environment. In IEEE IS, 2018.

**[Gabbasov et al., 2015]** B. Gabbasov, I. Danilov, I. Afanasyev, and E. Magid. Toward a human-like biped robot gait: Biomechanical analysis of human locomotion recorded by Kinect-based Motion Capture system, ISMA, 2015.

**[Ibragimov et al., 2017]** Ibragimov I. et. al. Comparison of ROS-based visual slam methods in homogeneous indoor environment. In WPNC, 2017.

**[Khusainov et al., 2016]** Khusainov R., Shimchik I., Afanasyev I. and Magid E. 3D modelling of biped robot locomotion with walking primitives approach in Simulink environment. LNEE (383): 287-304. Springer, 2016.

**[Lavrenov et al., 2017]** Lavrenov R., Zakiev A. Tool for 3D Gazebo Map Construction from Arbitrary Images and Laser Scans. In DeSE, 2017.

**[Magid et al., 2017]** Magid E., Lavrenov R., Afanasyev I. Voronoi-based trajectory optimization for UGV path planning. In ICMSC, 383-387, 2017.

**[Rohmer et al., 2013]** E. Rohmer, S.P. Singh, M. Freese. V-REP: A versatile and scalable robot simulation framework. In IEEE IROS, 1321-1326, 2013.

**[Sheikh et al., 2018]** Sheikh T.S., Afanasyev I.M. Stereo vision-based optimal path planning with stochastic maps for mobile robot navigation. Advances in Intelligent Systems and Computing, vol 867, 40-55, Springer, 2018.

**[Shimchik et al., 2016]** Shimchik I., Sagitov A., Afanasyev I., Matsuno F., Magid E. Golf cart prototype development and navigation simulation using ROS and Gazebo. In MATEC Web of Conferences, vol. 75, p. 09005, 2016.

**[Sokolov et al., 2016]** Sokolov M. et al. 3D modelling and simulation of a crawler robot in ROS/Gazebo. In ICMA, ACM, 61-65, 2016.

**[Sokolov et al., 2017a]** Sokolov M. et al. Modelling a crawler-type UGV for urban search and rescue in Gazebo environment. In Proc. ICAROB, 2017.

**[Sokolov et al., 2017b]** Sokolov M. et al. HyperNEAT-based flipper control for a crawler robot motion in 3D simulation environment. In ROBIO, 2017.

**[Stanley et al, 2002]** K.O. Stanley, R. Miikkulainen. Efficient evolution of neural network topologies. In IEEE CEC, vol. 2, 1757-1762, 2002.

**[Zubov et al, 2018]** Zubov I., Afanasyev I., Shimchik I., Mustafin R., Gabdullin A. Autonomous Drifting Control in 3D Car Racing Simulator. In IEEE Intelligent Systems (IS), 2018.