ВЕСТНИК ВОЕННОГО ИННОВАЦИОННОГО ТЕХНОПОЛИСА «ЭРА», 2022, том 3, № 2, с. 175–181

МОБИЛЬНЫЕ КОМПЛЕКСЫ И РОБОТОТЕХНИЧЕСКИЕ СИСТЕМЫ

УДК 004.896:621.865

РАЗРАБОТКА СРЕДСТВ ОБУЧЕНИЯ РЕАЛИСТИЧНЫХ МОДЕЛЕЙ МОБИЛЬНЫХ РОБОТОВ В СИМУЛЯТОРЕ ROS GAZEBO

© 2022 г. А. Д. Московский[1],*, М. А. Ровбо[1]

[1]Национальный исследовательский центр "Курчатовский институт", Москва, Россия

***E-mail: Moskovsky_AD@nrcki.ru**

Поступила в редакцию 15.03.2022 г.
После доработки 20.03.2022 г.
Принята к публикации 20.03.2022 г.

Рассмотрена проблема сбора данных для обучения нейросетевой модели на примере создания системы безопасности для роботизированной инвалидной коляски. Важным аспектом этой проблемы является необходимость работы с данными, получаемыми на разных уровнях детализации: от упрощенного прототипа модели до реального робота. Для ее решения предложена архитектура системы сбора данных и разработано соответствующее программное обеспечение. Созданные алгоритмы апробированы на задаче обучения нейросетевой модели предсказанию положения робота и вероятности попадания его в опасные ситуации на основе сенсорных данных. Использование в реализации интерфейсов фреймворка ROS позволило протестировать систему как на модели в симуляторе Gazebo, так и на реальном роботе. Собранные в процессе работы системы данные использованы в стандартных алгоритмах обучения для получения предварительных результатов.

DOI: 10.56304/S2782375X22020127

ВВЕДЕНИЕ

Робототехника всегда была тесно связана с областью искусственного интеллекта. Благодаря этому развитие в последние годы глубокого машинного обучения также повлияло на робототехнику. Обучаемые нейросетевые модели широко используются в задачах распознавания и предсказания. Одной из трудностей в применении глубоких моделей является процесс получения репрезентативной обучающей выборки. В некоторых задачах допустимо использовать неразмеченную выборку [1, 2]. Однако в большинстве случаев требуется разметка данных, которая часто проводится вручную. Это трудоемкий и времязатратный процесс, но альтернативой является только автоматическая разметка данных, которая в реальных условиях не всегда возможна из-за отсутствия нужной информации. Также сбор данных можно проводить в симуляторах, в которых всегда доступна полная информация об агентах и среде [3]. Однако искусственно синтезированные данные не всегда соответствуют реальным, поэтому требуется создание средств синтеза обучающих данных, максимально приближенных к реальным, а также валидации этих данных и обученных на них моделей.

Разработка средств обучения показана на примере задачи создания системы безопасности для роботизированной инвалидной коляски [4]. Роботизированная инвалидная коляска является полигоном для отработки различных альтернативных способов управления для людей с сильными повреждениями верхних конечностей, в том числе с использованием технологии отслеживания взгляда [5]. Проблема такого подхода заключается в том, что система не всегда способна отличить спонтанное движение глаз от осознанной команды, что может приводить к движению в случаях, когда пользователь этого не хочет. Иногда это приводит к опасным для пользователя и окружающих ситуациям. Можно решать такую проблему двумя способами. Первый – использование другого канала передачи информации для подтверждения команды: голоса, мио- и тактильных интерфейсов, а также электроэнцефалограммы [6]. Второй способ – интеллектуализация системы управления, чтобы она сумела на основе данных от сенсоров проанализировать команду пользователя и заблокировать ее, когда может возникнуть опасная ситуация. При движении по ровной поверхности (комнатные условия, двумерная задача) такая система безопасности реализуется экстраполяцией траектории робота и определением пересечения его проекции с двумерной картой препятствий, полученной по данным сканирующего лазерного дальномера и/или камеры глубины. Однако когда заходит речь об использовании в уличных условиях, задача более не может быть двумерной: появляются такие объ-

екты, как провалы (тротуары, лестницы, открытые канализационные люки) и наклонные поверхности (пандусы, неровное покрытие, холмистая местность). При движении по данным объектам есть риск как падения робота с высоты, так и его опрокидывания. Экстраполяция траектории робота при движении по произвольной трехмерной поверхности становится более трудоемкой. Вместо того чтобы решать эту задачу аналитически, можно собрать набор данных о том, как проходит движение по такой поверхности, и обучить нейросетевую модель генерировать предсказания положения робота, а также вероятности попадания в опасные ситуации на основе имеющихся данных от сенсоров.

Для обучения такой системы важной является проблема зависимости эффективности работы алгоритма от качества собранных данных. Зачастую разработка подобных систем происходит поэтапно. Сначала разрабатывается прототип с более простой структурой обучаемого алгоритма, упрощенной моделью робота и взаимодействия со средой. В финальной версии происходит переход к реалистичной модели с практическими сценариями использования и сбором данных, совместимым с реальным роботом. Поэтому средства сбора данных и обучения должны поддерживать соответствующие модификации в ходе разработки.

Современные библиотеки и фреймворки машинного обучения, как правило, хорошо решают задачу стыковки алгоритмов обучения с широким классом задач путем использования стандартных интерфейсов, под которые разработчики могут создавать свои среды для конкретных задач. Например, StableBaselines [7] использует OpenAIGym [8]. При этом привязка интерфейсов к средам поддерживается в основном для чисто программных сред (таких как Atari [9]) и симуляторов (например, MuJoCo [10]). Другие библиотеки [11] направлены на то, чтобы связать робототехнические интерфейсы с алгоритмами обучения. Однако в силу множества особенностей задач, на решение которых они направлены, создание универсального фреймворка затруднено. Многие из них перестают поддерживаться разработчиками в достаточной мере для использования с обновляемыми сторонними библиотеками и широкого распространения не получают. Поэтому для решения поставленной задачи применялся комбинированный подход, в котором сбор данных и проведение экспериментов в симуляции и на реальном роботе осуществляются разработанной конкретно под нее модульной системой, а для обучения и дальнейшей отладки обеспечивается совместимость с интерфейсами соответствующих библиотек машинного обучения, в частности с библиотекой PyTorch.

Таким образом, сформулированы следующие требования к разрабатываемой системе:

– возможность заменять алгоритм поведения агента при сборе данных, структуру обучаемого алгоритма, модель робота и среды;

– совместимость интерфейсов, используемых при сборе данных в модели и на реальном роботе;

– возможность использования реалистичных датчиков, включая шумы и настройку параметров;

– совместимые с реальным роботом средства ручной работы с данными для отладки процесса.

Для разработки системы сбора и обработки данных, удовлетворяющей этим требованиям, выбрана модульная структура, использующая интерфейсы на основе фреймворка ROS, совместимые с симулятором Gazebo и соответствующим роботом.

МЕТОДЫ

Для решения поставленной задачи разработали нейросетевую модель (рис. 1), позволяющую предсказывать вероятность класса события по данным от стереокамеры, текущим и желаемым скоростям.

Как было упомянуто, модель имеет три входа:

– последовательность из L наборов желаемых скоростей;

– стереоизображение, о структуре которого подробнее расскажем далее;

– текущие скорости робота.

Последняя размерность векторов скоростей равна двум, так как учитывается и линейная, и угловая скорости робота. Желаемая (заданная в команде) и текущая скорости подаются на вход многослойных перцептронов (MLP1 и MLP2). Перцептроны состоят из повторяющихся блоков, каждый из которых включает в себя полносвязный слой, dropout и функцию активации ReLU. Количество блоков и число нейронов в слоях – настраиваемые параметры. Для работы со стереоизображением используется распространенный предобученный сверточный энкодер MobileNetv3 [12] с отключенными слоями нормализации серий (BatchNorm). Отключение BatchNorm требуется, так как в отличие от цветных изображений стереоизображение содержит величины реальных расстояний, которые не могут быть изменены. Выходы энкодера и перцептрона текущей скорости соединяются и подаются на рекуррентный блок GRU в качестве начального скрытого слоя. Выход перцептрона желаемой скорости подается L раз на вход блока GRU с получением L выходов, которые функцией активации softmax трансформируются в вероятности каждого класса (если

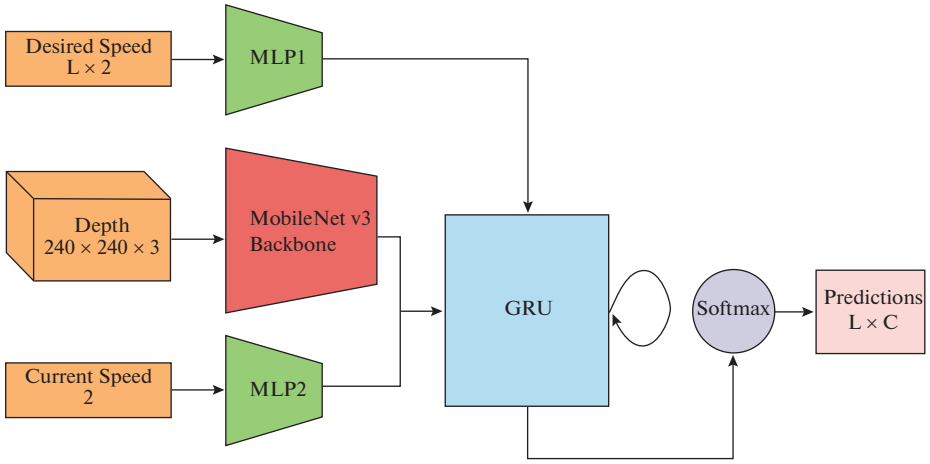

Рис. 1. Архитектура нейросетевой модели.

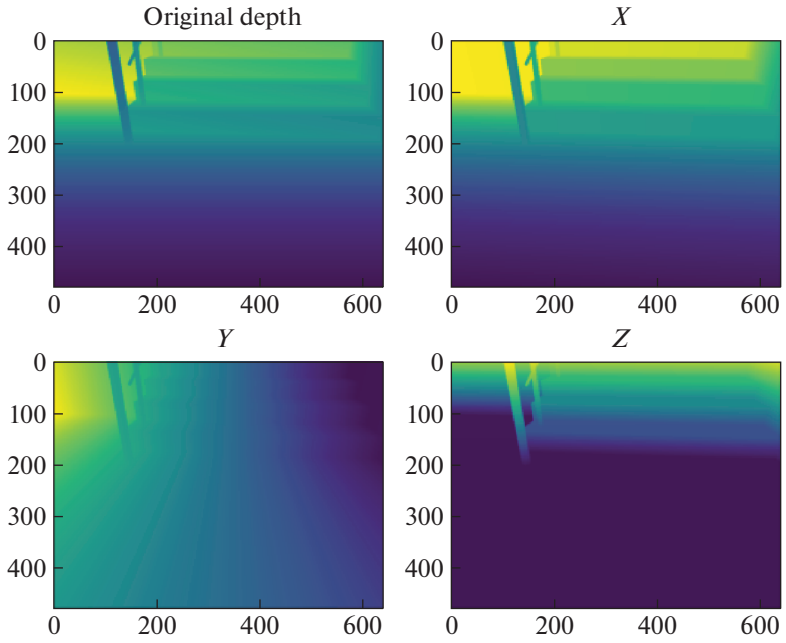

Рис. 2. Перевод стереоизображения в другую систему координат и разложение по осям.

классов всего два, то используется функция sigmoid).

Стереоизображение обычно представляет собой одноканальное изображение, где каждый пиксель отвечает за реальное расстояние до соответствующей ему точки пространства. Такое представление не содержит ни информации о расположении камеры, ни ее характеристик (фокусного расстояния, угла обзора и т.п.). Также большинство предобученных энкодеров могут работать только с трехканальным входом. Обе проблемы решаются путем перевода стереоизображения в другую систему координат и его разложения на три составляющих:

$$D_{orig} = \{d_{ij} \vee i \in [0, W], j \in [0, H]\},$$

$$X = \{x_i = (i - c_x)/f_x \vee i \in [0, W]\},$$

$$Y = \{y_j = (j - c_y)/f_y \vee j \in [0, H]\},$$

$$XY = (X \otimes Y) \cdot D_{orig},$$

$$D_{proj} = \begin{bmatrix} XY \\ D_{orig} \\ 1 \end{bmatrix} \times T,$$

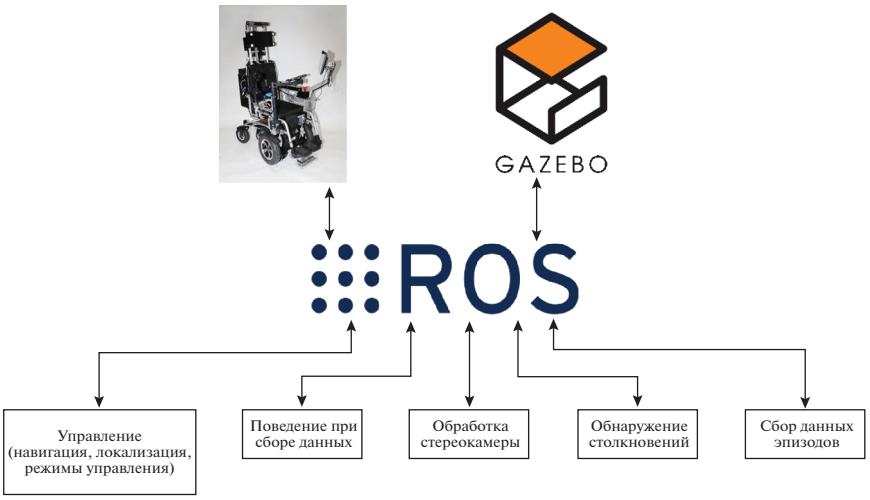

Рис. 3. Архитектура системы сбора данных.

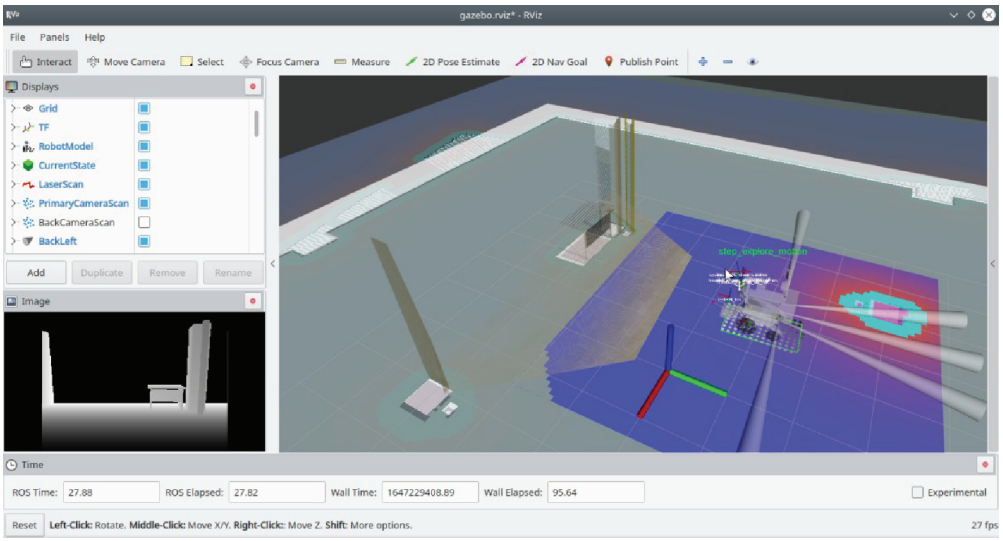

Рис. 4. Визуализация данных модели робота во время симуляции.

где D_{orig} – исходное изображение, f_x, f_y, c_x, c_y – параметры калибровки камеры, T – матрица трансформации из координат камеры в основную систему координат робота, D_{proj} – проецированная карта глубины. На рис. 2 изображены исходная и синтезированная карты глубины, разложенные по основным каналам.

Такое представление очень похоже на представление в виде облака точек, однако все еще представляет собой изображение, что позволяет использовать сверточные архитектуры без дополнительных преобразований [13]. Также такое представление наиболее устойчиво относительно небольших смещений положения камеры, что позволит регулировать ее положение.

Система сбора данных состоит из модели среды и робота, запускаемой в симуляторе Gazebo, программного обеспечения управления роботом, модуля обнаружения столкновений, модуля управления поведением агента при сборе данных, модуля сбора данных, модулей обработки сенсорных данных, позволяющих пользоваться симулированной стереокамерой и прочими датчиками модели (рис. 3).

При сборе данных в упрощенной модели робот использует камеру глубины (рис. 4) вместе с другими стандартными средствами обнаружения препятствий и навигации. При этом он передвигается в мире симулятора Gazebo (рис. 5), реализующего модель с учетом физики, что позволяет

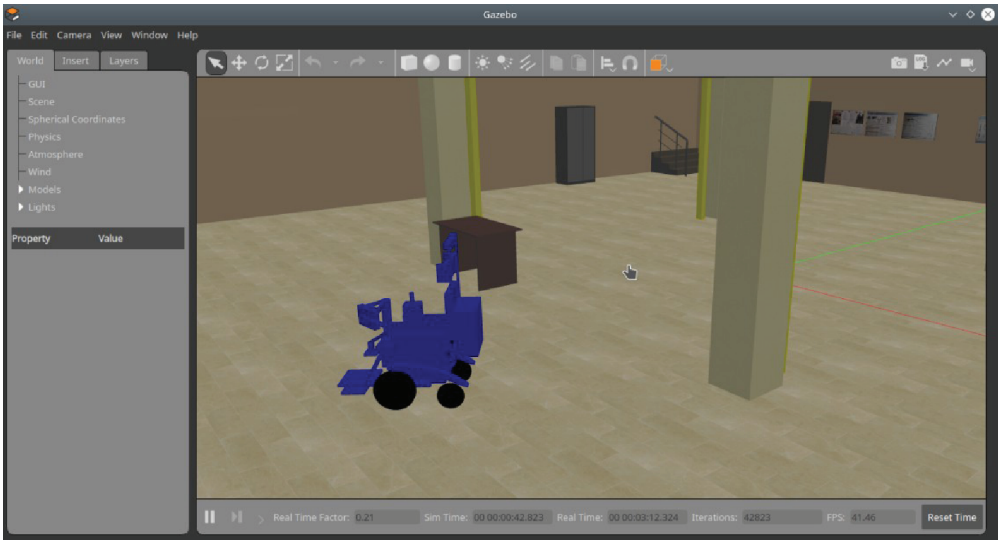

Рис. 5. Модель робота и среды в симуляторе Gazebo.

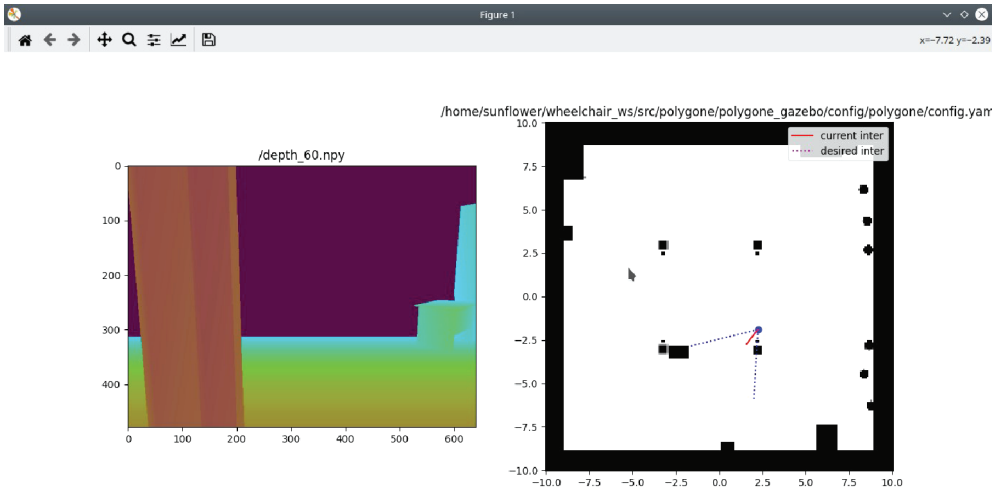

Рис. 6. Визуализация данных об эпизодах столкновения по собранным данным.

приближать такие эффекты, как инерциальность робота, проскальзывание колес и т.п.

Более близкая к реальному роботу модель использует вместо камеры глубины стереопару, проводящую обработку видеопотока и формирующую изображение глубины. С добавлением эффектов шума и настройкой реалистичных параметров углов обзора это позволяет обеспечить при обучении более простой переход к использованию алгоритмов на реальном роботе, а также к дообучению моделей благодаря совместимому интерфейсу. Успешность подобного подхода показана в [3].

При разработке для роботов основанных на обучении алгоритмов необходимо тщательно проверять адекватность входных данных и соответствие различных параметров предполагаемым. Помимо программных ошибок могут возникать проблемы с синхронизацией данных в силу параллельной работы множества датчиков с различной частотой, нередко ощутимой задержки в обработке данных (например, изображения), ограниченности вычислительной мощности компьютера, на котором проводятся моделирование и обработка данных датчиков, что может приводить к расхождению в ожидаемой и реальной частоте управляющего цикла.

Для отслеживания этих эффектов использовали комбинацию стандартных средств сбора логов о работе системы в ROS (rosbag, рис. 7) и разрабо-

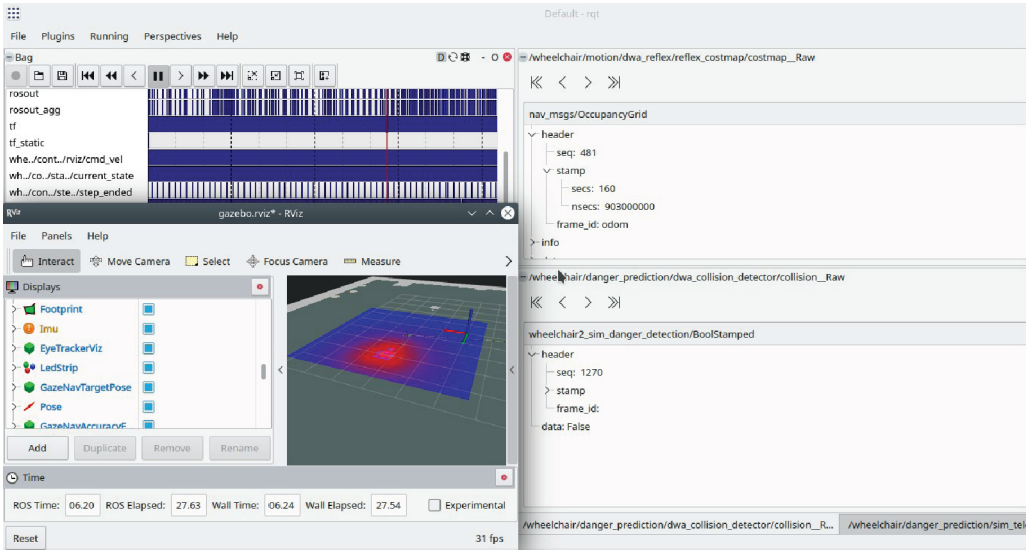

Рис. 7. Отладка собираемых данных с помощью воспроизведения rosbag-файла с экспериментом.

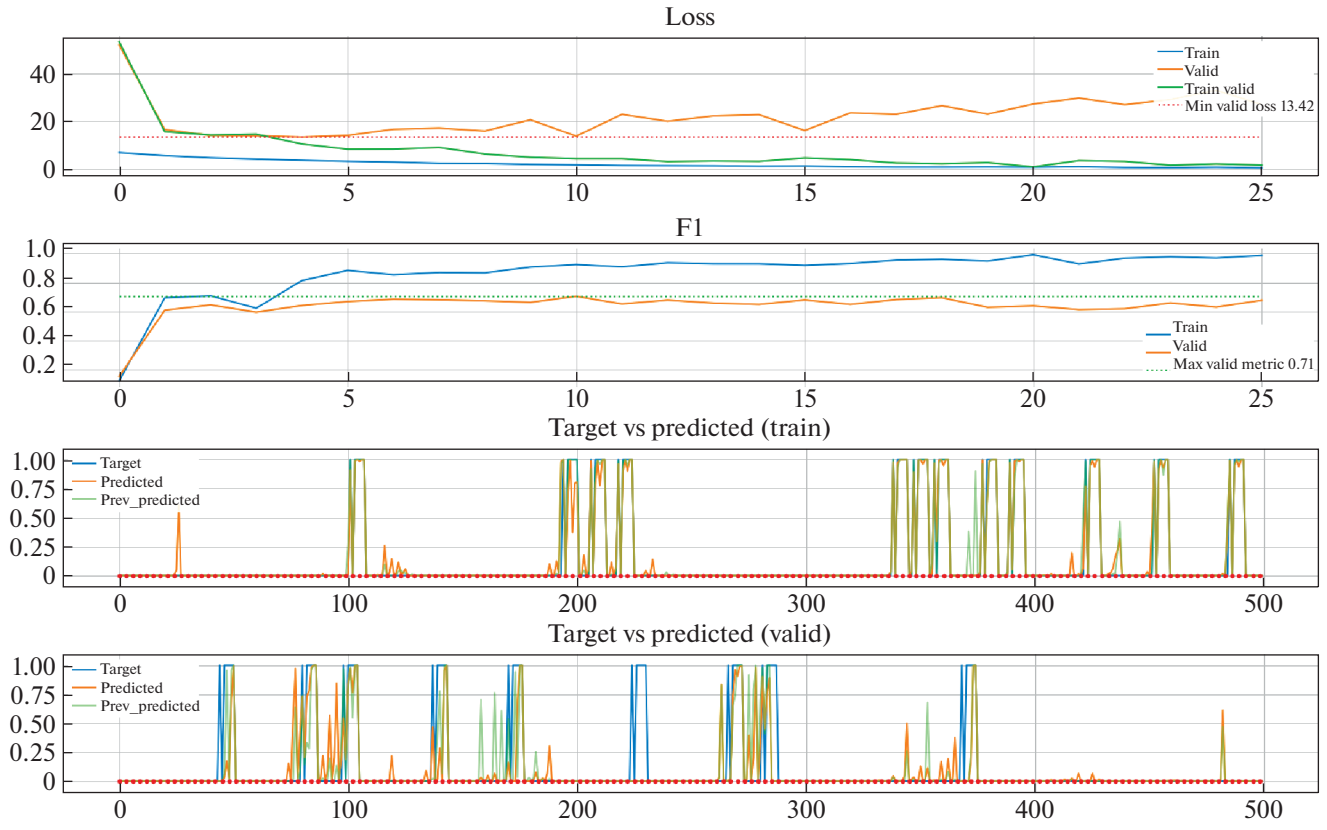

Рис. 8. Графики для отслеживания процесса обучения.

танных программ визуализации эпизодов и очистки данных (рис. 6).

Визуализация важных событий (обнаруженных столкновений) среди всего потока собираемых данных позволяет заметить такие проблемы, как смещение изображения глубин относительно положения робота в силу задержки обработки, обнаружение ложных столкновений, ошибки в выборе длины экстраполяции траектории во вре-

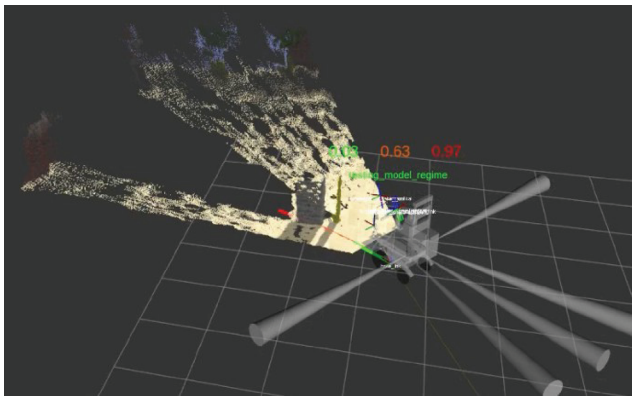

Рис. 9. Верификация модели.

мя эпизода, проблемы в обработке устаревших данных при начале нового эпизода после перемещения робота в новое положение и другие. Возможность повторного многократного воспроизведения эксперимента с замедлением времени и покадровой прокруткой с помощью инструментов rosbag и RViz также оказывается незаменимой в этом процессе.

РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

В процессе обучения также важно отслеживать различные параметры, для чего обучающий скрипт генерирует набор графиков, из которых можно получить достаточно информации о качестве получившейся модели. На рис. 8 изображены графики процесса обучения для сети с двумя классами ситуаций (штатная ситуация и опасность).

Первый график показывает динамику функции потерь на обучающей и тестовой выборке. Причем для обучающей выборки приведены два графика: для режима обучения и предсказания. Пунктирной линией отмечен момент минимального значения функции потерь для тестовой выборки, позволяющего определить момент начала переобучения сети. Второй график показывает валидационную метрику, в данном случае $F1$, для обучающей и тестовой выборки. На двух последних графиках изображены желаемые и предсказанные значения на последнем и предыдущем шаге (для оценки динамики) у обучающей и тестовой выборки. Эти графики позволяют увидеть характер предсказаний, а также локализовать проблемные моменты в выборках для дальнейшего анализа.

Финальная верификация модели происходит в среде симуляции. Робот перемещается в ручном режиме при помощи джойстика/геймпада, а в визуализаторе RViz отображаются вход и выход модели (рис. 9).

На рис. 9 изображен робот: его карта глубины в виде трехмерного облака точек; траектория робота, отображенная линией на уровне пола, раскрашенной согласно предсказанным значениям, которые также отображены над моделью робота. В данном случае траектория, состоящая из трех сегментов, ведет к столкновению с колонной, что обозначают предсказанные значения.

ЗАКЛЮЧЕНИЕ

Разработаны архитектура системы сбора данных и обучения мобильного робота для задачи автоматического обнаружения опасных ситуаций на примере столкновений, соответствующий набор программных средств сбора и визуализации данных, модель среды и робота для симулятора Gazebo. Все созданное программное обеспечение реализовано в соответствии с фреймворком ROS, что обеспечивает его совместимость с реальным роботом. Процесс сбора данных апробирован как в модели, так и на реальном роботе. Проведены тестовые эксперименты для проверки работоспособности системы сбора данных и эффективности процесса ее отладки. Система показала себя работоспособной и дала предварительные результаты по обнаружению столкновений, однако исследования по повышению эффективности системы обучения еще продолжаются.

Работа выполнена при поддержке НИЦ “Курчатовский институт” (приказ № 2754 от 28.10.2021).

СПИСОК ЛИТЕРАТУРЫ

1. *Wang S., Clark R., Wen H. et al.* // IEEE ICRA. 2017. P. 2043.
2. *Song Y., Tian Y., Wang G., Li M.* // IEEE ICRA. 2019. P. 6617.
3. *Loquercio A., Kaufmann E., Ranftl R. et al.* // Sci. Robot. 2021. V. 6. № 59. P. 5810.
4. *Karpov V.E., Malakhov D.G., Moscowsky A.D. et al.* // Sovrem. Tehnol. Med. 2019. V. 11. № 1. P. 90.
5. *Московский А.Д.* // Вестник военного инновационного технополиса Эра. 2021. Т. 2. № 3. С. 27.
6. *Чжао Д.Г. et al.* // Сборник материалов конференции “Когнитивная наука в Москве: Новые исследования – 2021.” 2021. С. 548.
7. Stable Baselines. https://github.com/hill-a/stable-baselines
8. OpenAIGym. https://github.com/openai/gym
9. *Bellemare M.G., Naddaf Y., Veness J., Bowling M.* // J. Artif. Intell. Res. 2013. V. 47. P. 253.
10. *Todorov E., Erez T., Tassa Y.*// IEEE IEEE Int. Conf. Intell. Robots Syst. 2012. P. 5026.
11. openai_ros http://wiki.ros.org/openai_ros.
12. *Howard A., Sandler M., Chu G. et al.* // 2019. arXiv: 1905.02244.
13. *Meyer G.P., Laddha A., Kee E. et al.* // Proc. IEEE Comput. Soc. Conf. Comput. Vis. 2019. P. 12677.