Системы управления интеллектуальных мобильных роботов в среде Dyn-Soft RobSim 5

Евстигнеев Д.В.

Москва, 2014

Оглавление

1. Введение

1.1.О чем данное учебное пособие

Данное учебное пособие предназначено для самостоятельной работы студентов-специалистов и студентов-бакалавров при подготовке дипломных и выпускных работ, а также студентов-магистров, изучающих устройство, алгоритмы и методы реализации различных систем интеллектуальных мобильных роботов. В учебном пособии рассказывается о реализации интеллектуальных систем управления, систем очувствления, систем навигации и информационно-измерительных систем автономных мобильных роботов. Кроме того, в учебном пособии приводится примеры реализации этих систем на базе учебного программного комплекса Dyn-Soft RobSim 5, цель которого, не вдаваясь в глубины программирования, создать виртуальную модель интеллектуального мобильного робота и его систему управления.

Если опустить главы, связанные с разработкой роботов в Dyn-Soft RobSim 5, то данное учебное пособие будет интересно также разработчикам реальных мобильных роботов, т.к. в данном учебном пособии подробно описаны различные устройства, входящие в состав системы очувствления интеллектуальных мобильных роботов, алгоритмы их работы, а также реализация различных интеллектуальных алгоритмов системы управления роботов. Поэтому работа с Dyn-Soft RobSim 5 вынесена в отдельные главы.

Данное учебное пособие подразумевает, что студенты, применяющие Dyn-Soft RobSim 5, уже ознакомились с предыдущими двумя учебными пособиями «Проектирование роботов и робототехнических систем в Dyn-Soft RobSim 5. Часть I» и «Проектирование роботов и робототехнических систем в Dyn-Soft RobSim 5. Часть II» и обладают навыками работы с данным программным комплексом. Поэтому в данном учебном пособии не уделяется большого внимания порядку разработки конструкции и электрических схем робота. Считается, что студент, решивший реализовать интеллектуального мобильного робота в Dyn-Soft RobSim 5, должен обладать навыками проектирования обычного дистанционно-управляемого робота, о порядке разработки которого шла речь в предыдущих учебных пособиях.

1.2. Общие сведения об интеллектуальных системах управления мобильных роботов

1.2.1. Понятие интеллектуальных мобильных роботов

Существует целый ряд задач, при которых применение дистанционно-управляемых роботов затруднено или невозможно. Например:

1. Невозможно организовать канал связи с роботом.
2. Имеется вероятность потери связи с роботом при его нахождении в недоступном для человека месте.
3. Выполняемая роботом задача не подразумевает ручного управления (роботы-уборщики, роботы-официанты, роботы-курьеры и т.п.).
4. Конструкция робота столь сложна, что человек не способен контролировать весь функционал робота.

Для решения этих задач в настоящее время активно ведутся разработки автономных мобильных роботов. В силу того, что такие роботы, должны работать в заранее неизвестных условиях, а ориентироваться во внешней среде на основании показания большого числа датчиков, применение классических математических подходов к таким системам оказывается невозможным.

В 60-годах прошлого века появилось направление, называемое *интеллектуальные системы*. Данные системы действуют не по заранее заданной программе, а самостоятельно принимают решение на основе информации об окружающей среде и заложенных в них знаниях. Функционирование таких систем в сложной и заранее неизвестной среде и в подавляющем числе случаев принимают верное решение, основанное на заложенной в них логике поведения.

Искушенный ум читателя уже стал представлять себе армию интеллектуальных роботов, желающих захватить мир. Однако следует понимать, что о создании искусственного разума речи не идет.

Интеллектуальной системой в технике называют системы, работающие не по жестко заданной программе (циклограмме), а принимают решение на основе заложенных в них правил. Интеллект таких роботов заключается в выполнении простейших (с точки зрения человеческого разума) задач, но, тем не менее, достаточно сложных в технической реализации. Например, классическое автономное движение мобильного робота к целевой точке в среде с препятствиями. Данная задача выполняется человеческим разумом на подсознательном уровне. Однако техническая ее реализация может быть достаточно сложной.

1.2.2. Классическая структурная схема интеллектуальной системы управления

Классическая структурная схема интеллектуальной системы управления мобильным роботом представлена на Рис. 1. Она включает в свой состав следующие структурные единицы:

- Нижний (приводной) уровень системы управления. Под приводным уровнем подразумевают регуляторы скорости и положения, реализованные внутри микропроцессоров контроллера управления исполнительными механизмами робота. На вход нижнего уровня системы управления поступают задания на скорости и углы поворота звеньев, а на выходе формируются управляющие напряжения для двигателей робота. В последние годы в связи с бурным ростом производительности микроконтроллеров и активным внедрением ПЛИС (программируемых логических матриц) стало модно разрабатывать для нижнего уровня системы управления интеллектуальные регуляторы, инвариантные к инерции нагрузки и самой нагрузке (в пределах мощности двигателя).
- Тактический уровень системы управления (система управления движением) представляет собой интеллектуальные алгоритмы, решающие задачи движения мобильного робота в среде с препятствиями, а также контурная система управления движением манипулятора робота (ПЗК/ОЗК), в т.ч. интеллектуальные системы обхода препятствий манипулятором робота. На вход тактического уровня системы управления поступают целевые точки как для системы управления движением, так и для манипулятора робота. На выходе – заданные скорости и углы ориентации звеньев.

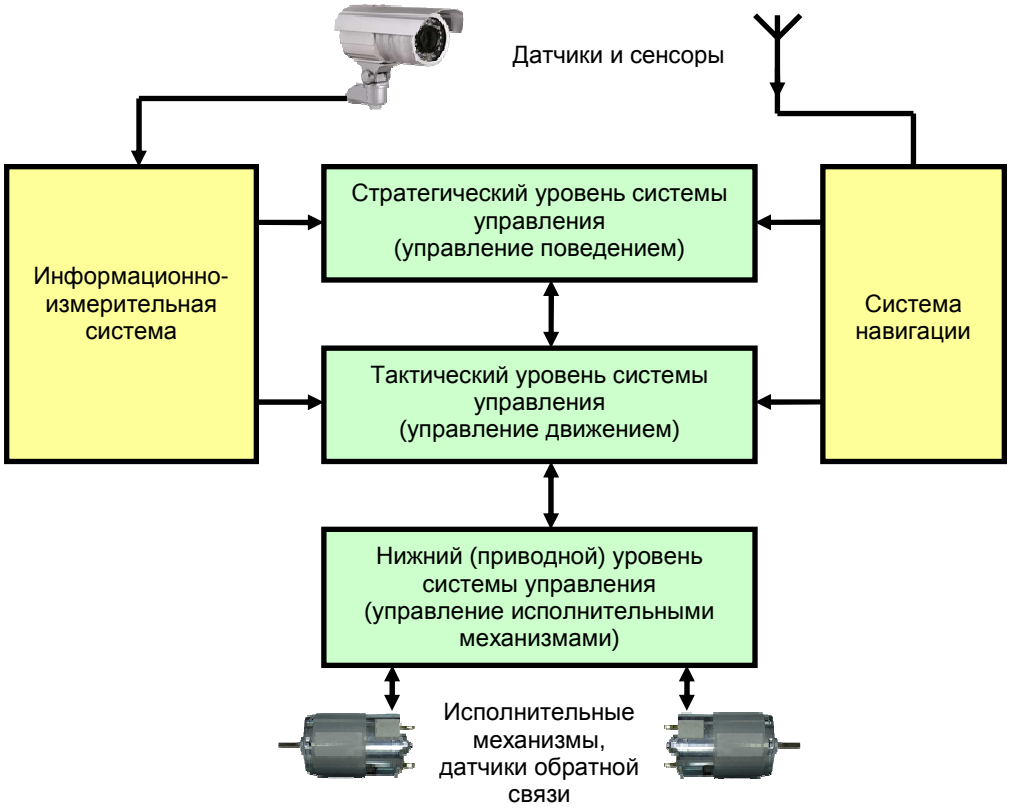

Рис. 1 Классическая структурная схема интеллектуальной системы управления мобильным роботом

- Стратегический уровень системы управления (система управления поведением) представляет собой интеллектуальную систему, позволяющая реализовать сложные последовательности действий робота, приводящие к решению им поставленной задачи. Стратегический уровень системы управления раскладывает поставленную задачу на более мелкие подзадачи до тех пор, пока она не будет состоять из заданий для тактического уровня системы управления.
- Система навигации позволяет роботу определять свое местоположение и ориентацию на карте местности. Данная информация необходима для выбора роботом направления движения к целевой точке
- Информационно-измерительная система позволяет роботу получать и обрабатывать информацию, полученную с различных сенсоров и датчиков. В ряде случаев, на основе обработки информации с нескольких датчиков удается получить новую информацию, например, на основе совместной обработки изображений с двух камер получить стереоизображение, на основе которого вычислить расстояния до препятствий.

Совокупность всех датчиков и сенсоров называют *системой очувствления*. К ней относятся видеокамеры, микрофоны, дальномеры,

акселерометры, гироскопы, магнитометры, некоторые датчики обратной связи (например, датчики на колесах).

Подробно о каждой из систем речь пойдет в следующих главах.

2. Система очувствления интеллектуальных мобильных роботов

2.1. Видеокамеры

2.1.1. Общие сведения о видеокамерах

Чувствительным элементом современных камер является ПЗС-матрица (CCD), хотя сейчас стали появляться КМОП-матрицы (CMOS).

Чем больше площадь пикселя матрицы (а, соответственно, самой матрицы), тем она более чувствительная к свету, и тем меньше в ее пикселях тепловых шумов. В принципе, именно наличие шумов не дает возможность бесконечно повышать чувствительность пикселя. Для борьбы с шумами не помогает даже большой светочувствительный объектив.

ПЗС-матрица работает по следующему принципу. Изначально происходит сброс заряда всех пикселей матрицы. Затем производится экспозиция. При этом под влиянием света в пикселях матрицы возникает заряд. Время засветки пикселей матрицы называется временем экспозиции. Чем больше время экспозиции, тем больший заряд накапливают пиксели (вплоть до насыщения), но тем больше на изображении проявляется смазывание от движения.

Современная технология не позволяет получить прямой доступ к заряду каждого отдельного пикселя. Поэтому заряд последовательно «сливают» с матрицы. Сначала созданием разности потенциалов заряд каждого пикселя перетекает от строки к строке. Затем из последней строки созданием разности потенциалов последовательно сливают заряд каждого пикселя в устройство регистратора заряда. У некоторых матриц может быть несколько выходов заряда. Такие матрицы, по сути, являются объединением нескольких матриц в одну большую.

Для видеокамер применяются матрицы с электронным затвором. Альтернативой электронному затвору является центральный механический затвор (как для фотоаппаратов). Электронный затвор реализуется за счет ввода в матрицу дополнительных теневых строк пикселей, закрытых от света. Во время экспозиции свет фиксируется в открытых для света пикселях, а затем по команде матрице заряд перетекает в закрытые пиксели, а уже оттуда последовательно строка за строкой, пиксель за пикселем «вытекает» из матрицы и регистрируется электронным устройством. Чувствительность матриц с электронным затвором ниже, чем

для матриц без электронного затвора, т.к. теневые пиксели уменьшают площадь всей матрицы.

Для цветных матриц применяется фильтр Байера (Рис. 2).

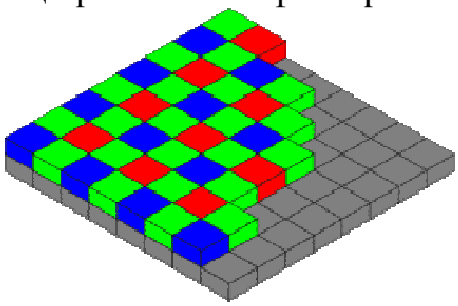

Рис. 2 Массив цветных фильтров Байера

Фильтр Байера предполагает наличие цветовых фильтров для красного, зеленого и синего цвета поверх пикселей черно-белой матрицы. Как видно из рисунка, каждый отдельный пиксель матрицы регистрирует только одну составляющую цвета (красную, зеленую или синюю). На конечном изображении цвет формируется из объединения цветов соседей. Существует заблуждение, что в «сыром» цветном изображении с матрицы в три раза больше информации, чем с аналогичной цветной. Это неправильно. Количество пикселей и объем информации с цветной и аналогичной черно-белой матрицы одинаково.

Кстати, обычно каждая матрица выпускается как в цветном, так и черно-белом варианте. Отличаются они лишь наличием фильтра Байера у цветных матриц.

При выборе камер, следует обратить внимание, что чувствительность черно-белых камер на много выше, чем у цветных. Поэтому, большинство профессиональных камер для технических, медицинских и астрономических исследований, а также охранные камеры – черно-белые.

2.1.2. Виды интерфейсов камеры

Существует несколько видов интерфейсов для передачи видеоизображения с видеокамеры на персональный компьютер:

- аналоговый видеосигнал формата PAL/SECAM/NTSC;
- USB;
- Ethernet (IP-камеры);
- WiFi;
- PCMCIA;
- LVDS/Camera Link;
- прочие.

PAL/SECAM/NTSC

Для захвата аналогового видеосигнала в формате PAL/SECAM/NTSC требуется специальная плата видеозахвата или TV-tuner. Следует отметить, что для передачи черно-белого изображения форматы PAL и SECAM похожи. Оба имеют стандартное разрешение 720x576 и 25 полных кадров в секунду. Кадр строится из четных и нечетных строк. Чередование четных и нечетных строк производится с частотой 50 Гц. На самом деле строк в видеосигнале 625, но часть строк из-за особенностей формирования видеосигнала не используется. Различия PAL и SECAM составляет способ кодирования цвета.

Имеются несколько разновидностей сигнала SECAM: SECAM-L, SECAM-K1, SECAM-B/G, SECAM-D/K. Все эти разновидности стандарта отличаются, в основном, способом передачи на высоких частотах (частотах телевизионного вещания) совместно с сигналом звукового сопровождения. А для передачи на низких частотах форматы почти ничем не отличаются. В телевизионном вещании в России применяется SECAM-D/K

Формат PAL имеет разновидности PAL-B, PAL-G/H, PAL-I, PAL-D/K, PAL-M, PAL-N. Разные разновидности формата PAL отличаются способом передачи на высоких частотах совместно с сигналом звукового сопровождения. Однако форматы PAL-M имеет 480 активных строк, а формат PAL-I – 582. В большинстве стран Западной Европы, Африки, Австралии и Южной Америки применяется стандарт PAL-B.

Формат NTSC имеет разрешение 704x480 и частоту 29.97 кадров в секунду. На самом деле в сигнале NTSC 525 строк, но часть строк из-за особенностей формирования видеосигнала не используется. Также сигнал NTCS использует черезстрочную развертку. Формат NTSC применяется в основном в США, Канаде и Японии. Есть две разновидности стандарта NTSC: NTSC-M и NTSC-J (для Японии).

Современные TV-тюнеры различают все варианты аналогового сигнала и способны их оцифровывать.

Следует отметить, что аналоговые камеры не обладают высоким разрешением. Изображение, оцифрованное с аналоговых камер, обладает помехами и нелинейными искажениями яркости, что не дает им особых преимуществ по сравнению с цифровыми камерами.

В ряде случаев аналоговые камеры могут быть совмещены с поворотным устройством, а также системой управления ZOOM и фокусом. Управление этими устройствами осуществляется по RS-485, в основном, по протоколу PELCO.

USB-камеры

USB-камеры – это, в основном, WEB-камеры. Однако, существуют и современные профессиональные камеры с USB-интерфейсом. Эти камеры

обладают не малым разрешением и высокой чувствительностью. В силу применения в них технологии JPEG-сжатия изображения, удается передать по USB 2.0 изображения высокого разрешения.

Следует отметить, что разработчики драйверов для USB-камер производят драйвера, в основном, для операционной системы Windows, в то время, как на бортовой ЭВМ робота рекомендуется использовать операционную систему Linux.

В Linux есть универсальные драйвера V4L (Video For Linux) и V4L2, применяемые для всех поддерживаемых Linux камер и TV-тюнеров. Поэтому, если USB-камера поддерживается этим драйвером, то проблем ее использования под Linux не будет. Если же поддержки данной камеры нет, то у разработчика могут возникнуть большие проблемы с подключением камеры к данной операционной системе. Причем, сложно дать какой-то определенный совет, какие конкретно модели камер поддерживаются в Linux: ни разработчики драйвера, ни разработчики камеры, обычно, этого не сообщают.

Следует отметить, что в последнее время наметилась тенденция стандартизировать драйвера USB-камер, используя стандарт UVC (USB video device class), который поддерживается как Windows, так и Linux.

Также некоторые USB-камеры имеют встроенное поворотное устройство. Для управления этим устройством следует использовать специальные функции драйвера.

IP-камеры

IP-камеры или Ethernet-камеры могут работать по протоколу: HTTP (JPEG), HTTP (MJPEG), RTSP/RTMP, GigE Vision или иметь иной специальный протокол.

Большинство IP-камер имеют WEB-интерфейс для своей настройки. Такие камеры являются, по сути, небольшими WEB-серверами. Поэтому для получения изображения с них применяется протокол HTTP.

В некоторых камерах для получения кадра видеоизображения следует запросить специальный (описанный в инструкции) документ с камеры по протоколу HTTP. В этом случае камера выдает один кадр изображение в формате JPEG. Для получения следующего кадра документ следует запросить заново.

В некоторых камеры в ответ на HTTP-запрос документа выдают изображение в формате MJPEG. В этом случае на один HTTP-запроос камера возвращает видеопоток из JPEG-изображений, завернутых в формат HTTP-ответов.

В принципе, для создания универсального драйвера IP-камеры достаточно, игнорируя все заголовки искать, в видеопотоке начало и конец JPEG-файла (JPEG имеет стандартное двухбайтное начало и стандартный двухбайтный конец).

Некоторые IP-камеры имеют интерфейс управления поворотом камеры, а также управления ZOOM. Для выполнения команды поворота следует запросить с камеры определенный документ по HTTP-протоколу. Запрос этого документа приводит к выполнению команды поворота.

Если камеры поддерживают RTSP/RTMP-протокол, то в Linux, скорее всего, разработчику придется писать собственную реализацию RTSP-плеера. В Windows имеются специальные плагины (фильтры) для DirectShow, позволяющие легко подключаться к RTSP/RTMP-потоку.

GigE Vision – стандарт используемый некоторыми камерами высокого разрешения (в основном черно-белыми). Несмотря на то, что стандарт предусматривает JPEG-сжания, ни одна известная камера его не использует. Все камеры согласно данному стандарту отправляет «сырое» видео по UDP (требуется Ethernet 1Гбит). Имеется несколько библиотек (Pleora SDK, JAI SDK, Aravis и др.) для захвата и управления GigE-камерами. Несложно также разработать собственный драйвер. Обмен данными производится по UDP, а описание стандарта выложено в Интернет. Разработка собственного драйвера осложняется тем, что различные камеры используют для управления различные адреса регистров. Для определения адресов регистров необходимо предусмотреть в драйвере скачивание с камеры специального XML-документа, сжатого архиватором ZIP, с последующим его разбором. Разбор данного документа осложняется тем, что для вычисления значений некоторых регистров применяются формулы, описанные в том же формате.

WiFi-камеры

WiFi-камеры представляют собой IP-камеры с совмещенным WiFi-роутером. Они могут работать как в режиме WiFi-клиент, так и в режиме ADHOC (WiFi-соединение точка-точка).

PCMCIA

Устаревший интерфейс подключения различных внешних устройств, карт памяти и камер. Использовался в основном для камер карманных компьютеров (КПК), а также для ноутбуков эпохи 2000-х годов.

LVDS/Camera Link

LVDS (low-voltage differential signaling) – низковольтная дифференциальная передача цифровых сигналов. Данный протокол известен также под названием RS-644. Имея амплитуду сигнала 350 мВ, протокол позволяет передавать до 1,923 Гбит/сек.

CameraLink – это интерфейс для высокоскоростной передачи видеоинформации, представляющий собой 4 LVDS-пары для передачи данных, 1 LVDS-пара для синхронизации, и 1 проводник – «земля».

По каждой LVDS-паре передаются последовательные 7-битные посылки, что позволяет передать по 4 линиям 28 бит данных.

Имеются три варианта реализации CameraLink:

- Base. Одна микросхема Camera Link, один разъем для кабеля.
- Medium. Две микросхемы Camera Link, два разъема для кабеля.
- Full. Три микросхемы Camera Link, три разъема для кабеля.

Во всех вариантах реализации один канал Camera Link используется для управления камерой, а остальные каналы используются для передачи видео. В варианте Base информация предается порциями по 28 бит, в варианте Medium – по 56 бит, в варианте Full – по 84 бита. Однако программная часть протокола в варианте Base передает 3 байта информации, в варианте Medium – 6 байт, а в варианте Full – 8 байт. Неиспользуемые биты считаются служебными и резервными. Максимальная скорость передачи данных в режиме Full – 680 Мбайт/сек.

Для захвата сигнала в формате CameraLink требуются специальные фреймграбберы.

2.1.3. Взаимная синхронизация камер

Для некоторых режимов работы системы технического зрения робота (например, для стереозрения) необходимо наличие взаимной синхронизации камер.

Под взаимной синхронизацией камер поднимают синхронизацию их затворов с точностью до микросекунды.

Для алгоритмов стереозрения синхронизация необходима для получения стереоизображения в движении или стереоизображения подвижных объектов. Дело в том, что даже при съемке со скоростью 25-30 кадров в секунду каждая камера фиксирует кадр в разные моменты времени. За время между моментами съемки с первой и второй камеры может переместиться как сам робот, так и объекты на сцене. Таких перемещений достаточно, чтобы полностью нарушить функциональность алгоритма стереозрения, т.к. он требует наличия двух изображений с разных точек одного и того же объекта.

Взаимная синхронизация может осуществляться несколькими способами. Наиболее простым является синхронизация по сети 50 Гц или синхронизация с помощью кабеля. Но для синхронизации необходимо, чтобы камера поддерживала данную технологию.

Следует отметить, что поддержка синхронизации имеется только у дорогих профессиональных камер. Для синхронизации с помощью кабеля нужно, чтобы одна из камер выступала в роли мастера и формировала специальные синхроимпульсы, а другие камеры выступали бы в роли ведомых камер.

На Рис. 3 и Рис. 4 показано устройство формирователя и приемника синхроимпульсов камеры USB uEye SE.

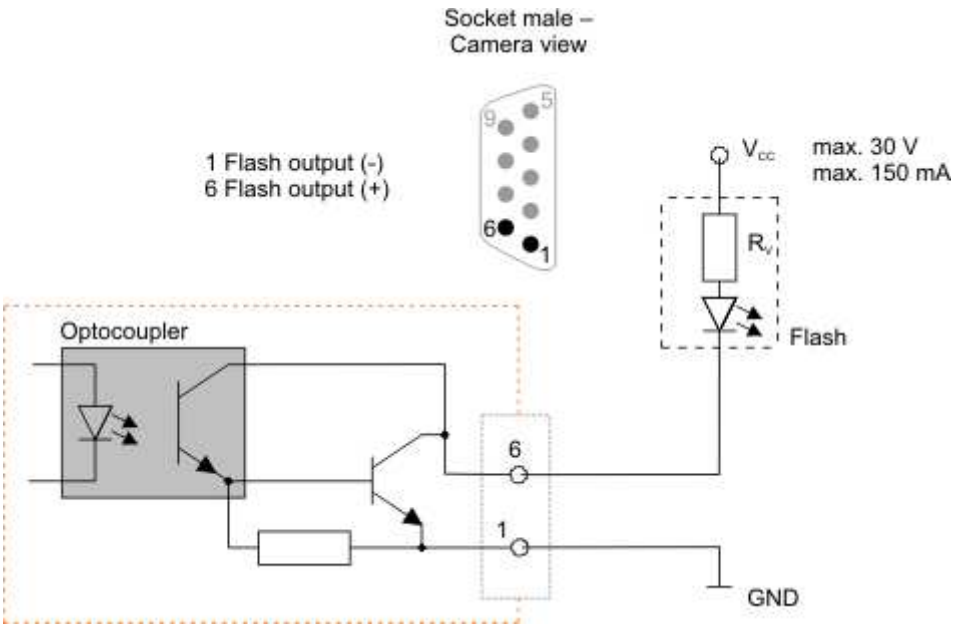

Рис. 3 Устройство формирователя синхроимпульсов камеры USB uEye SE

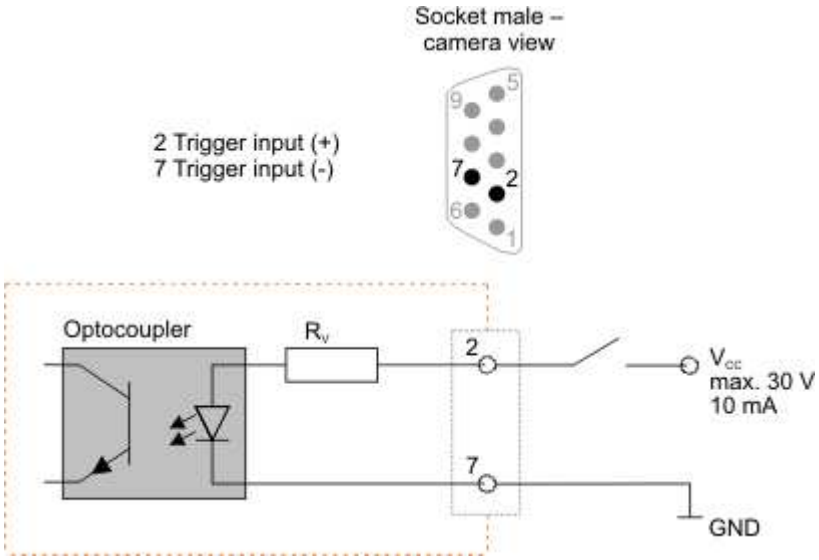

Рис. 4 Устройство приемника синхроимпульсов камеры USB uEye SE

Несмотря на то, что на рисунке данная схема позиционируется как система синхронизации со вспышкой, производитель подчеркивает возможность работы в режиме «Матер» и «Ведомый» с точностью синхронизацией до 1 мкс.

Следует обратить внимание, во-первых, в схеме используется оптоэлектронная развязка, поэтому «Земля» линии синхронизации не связана с «Землей» самой камеры. Во-вторых, выход линии синхронизации

представляет собой схему типа «Открытый коллектор», т.к. линия синхронизации требуется подтяжки к $+E_{пит}$. Производитель утверждает, что минимальное напряжение питания подтяжки +9В. Максимальное – +30В.

При организации подтяжки к $+E_{пит}$ следует обеспечить для выходного каскада линии синхронизации камеры ток не более 15 мА. С учетом внутреннего резистора у оптрона входного каскада в 3 кОм, при питании от +12В, требуется резистор подтяжки порядка 1 кОм.

2.1.4. Видеокамеры в среде Dyn-Soft RobSim 5

В среде Dyn-Soft RobSim 5 представлены видеокамеры, имеющие интерфейс PAL/SECAM, USB, Ethernet и WiFi. Для взаимной синхронизации камер поддерживается только синхронизация с помощью кабеля.

Установка камеры осуществляется в 3D Studio MAX (Рис. 5). Камера состоит из объекта «Камера» Dyn-Soft RobSim 5 и привязанного к нему объекта «Camera» 3D Studio MAX. Важно, никогда не разрушать этой привязки. Одной из типовых ошибок является выделение всех объектов и их привязка к какому-то другому объекту – при этом камера 3D Studio MAX отвязывается от своего родителя и привязывается к указанному объекту, т.е. привязка разрушается.

Объект «Камера» Dyn-Soft RobSim необходимо привязать к месту крепления камеры (если камера жестко связана с корпусом, то это необязательно). Если основание камеры выходит за пределы робота, то необходимо, во-первых, нарисовать в 3D Studio MAX объект, внешне похожий на камеру, а, во-вторых, аппроксимировать камеру объектом «Тело» с толщиной оболочки 0, так, чтобы массы у этого объекта не было, а поверхность столкновения была.

В зависимости от реализации источника питания для конкретной камеры, в редакторе схем и подключений Dyn-Soft RobSim 5 (кнопка «Схема») на камеру следует подать напряжение питания. Некоторые камеры питаются от USB, некоторые через PoE. В последнем случае следует установить через редактор схем и подключений инжектор питания PoE и запитать камеру требуемым напряжением.

Затем камеру требуется подключить к бортовой ЭВМ. Бортовая ЭВМ должна обладать соответствующим интерфейсом для подключения камеры, например, USB (Рис. 6).

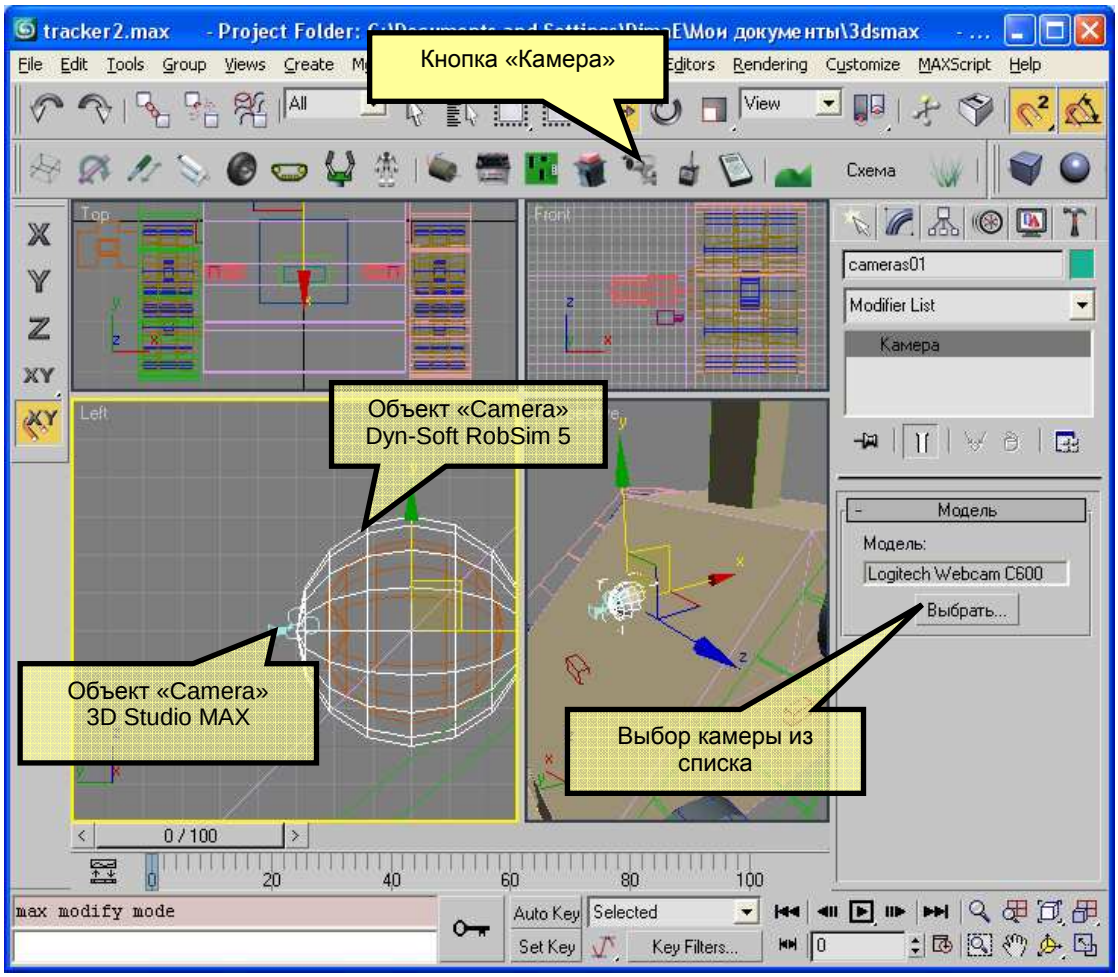

Рис. 5 Иллюстрация установки объекта «Камера» в 3D Studio MAX

Важную роль играют свойства камеры, открываемые в параметрах настройки камеры в редакторе схем и подключений (Рис. 7). Через него можно настроить канал для передачи, а также, опционально для приема управляющих данных, а также увидеть перечень передаваемых камерой параметров. В частности, камера, свойства которой показаны на рисунке передает по каналу 0 видеоданные и аудиосигнал.

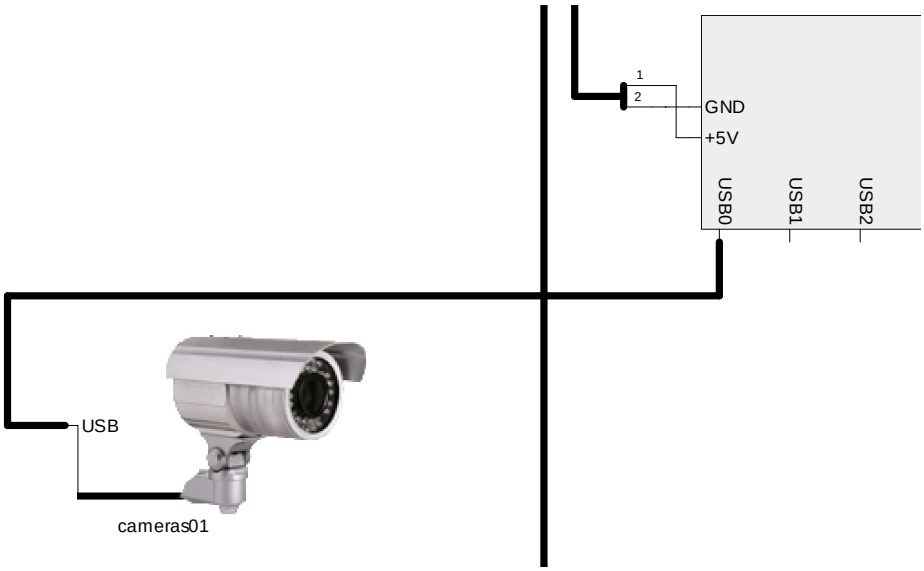

Рис. 6 Пример подключения камеры к USB-порту в редакторе схем и подключений Dyn-Soft RobSim 5

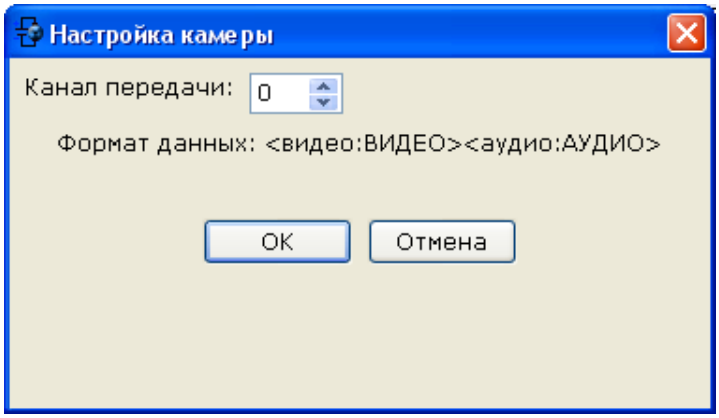

Рис. 7 Внешний вид диалога настройки камеры

В редакторе структурной схемы программного обеспечения для бортовой ЭВМ с интерфейса, к которому подключена камера, можно принять соответствующий сигнал для его дальнейшей передачи на пульт управления или для интеллектуальной обработки (Рис. 8).

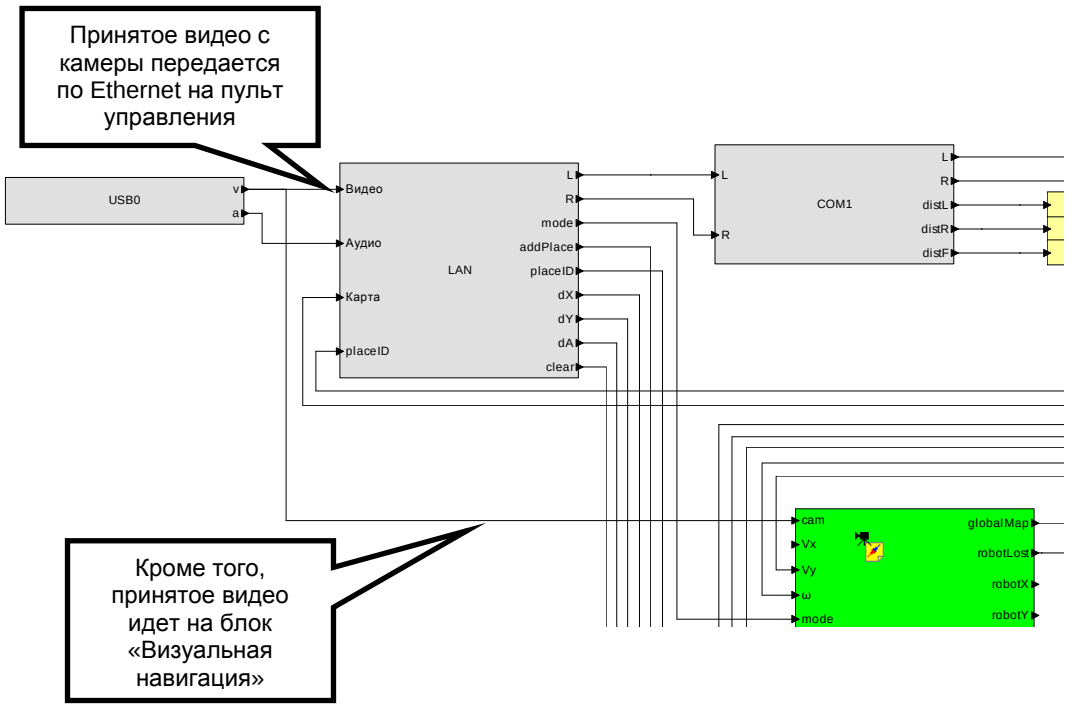

Рис. 8 Пример приема и передачи видео, а также его интеллектуальной обработки

2.1.5. Синхронизация камер в Dyn-Soft RobSim 5

Для синхронизации камер в Dyn-Soft RobSim 5 следует использовать схему, изображенную на Рис. 9.

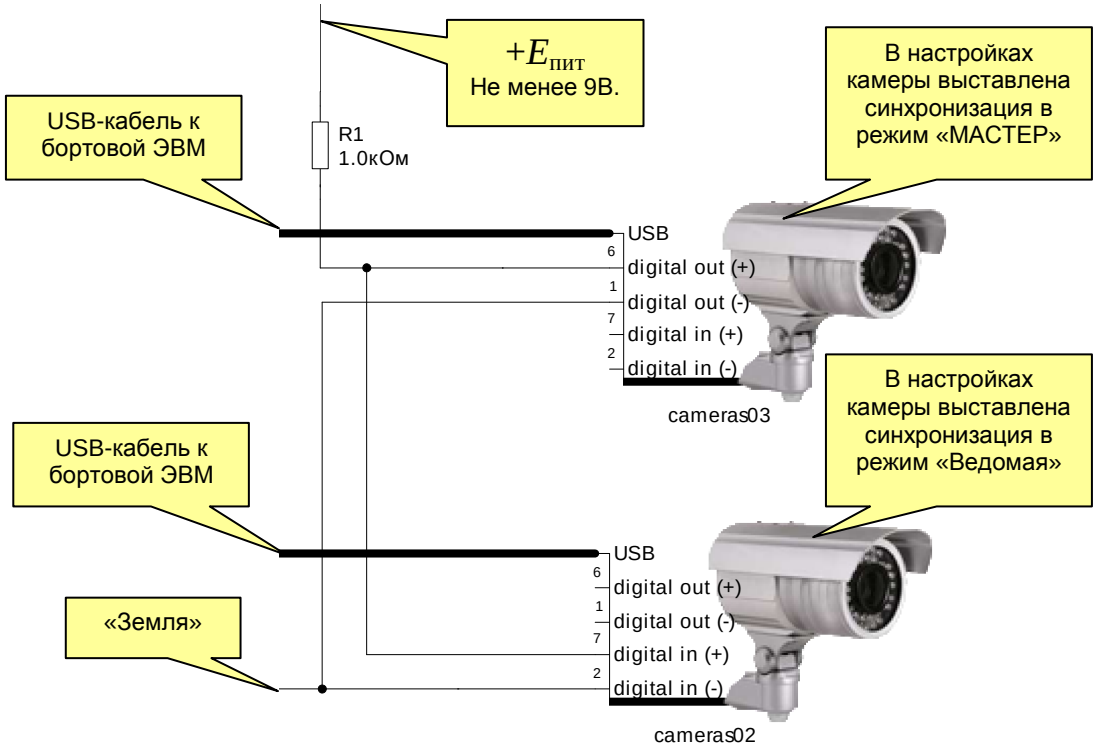

Рис. 9 Схема синхронизации камер в Dyn-Soft RobSim 5

Для синхронизации следует использовать только те камеры, в которых предусмотрен интерфейс синхронизации, например камеры USB uEye SE.

2.2.Ультразвуковые дальномеры

2.2.1. Принцип работы ультразвуковых дальномеров

Ультразвуковой дальномер во время измерения формирует ультразвуковой импульс (обычно 40 кГц), звук отражается от препятствия на пути и принимается микрофоном (Рис. 10).

Электронное устройство дальномера регистрирует время Δt между моментом формирования сигнала динамиком и его приема микрофоном (по некоторому пороговому уровню). Расстояние до препятствия определяется по формуле:

$$\Delta t = \frac{D_1 + D_2}{c} \approx \frac{2D}{c} \approx 58[\text{мкс/см}] \cdot D[\text{см}]$$

$$D = \frac{\Delta t \cdot c}{2}$$

Где: c – скорость звука в воздухе (343 м/с).

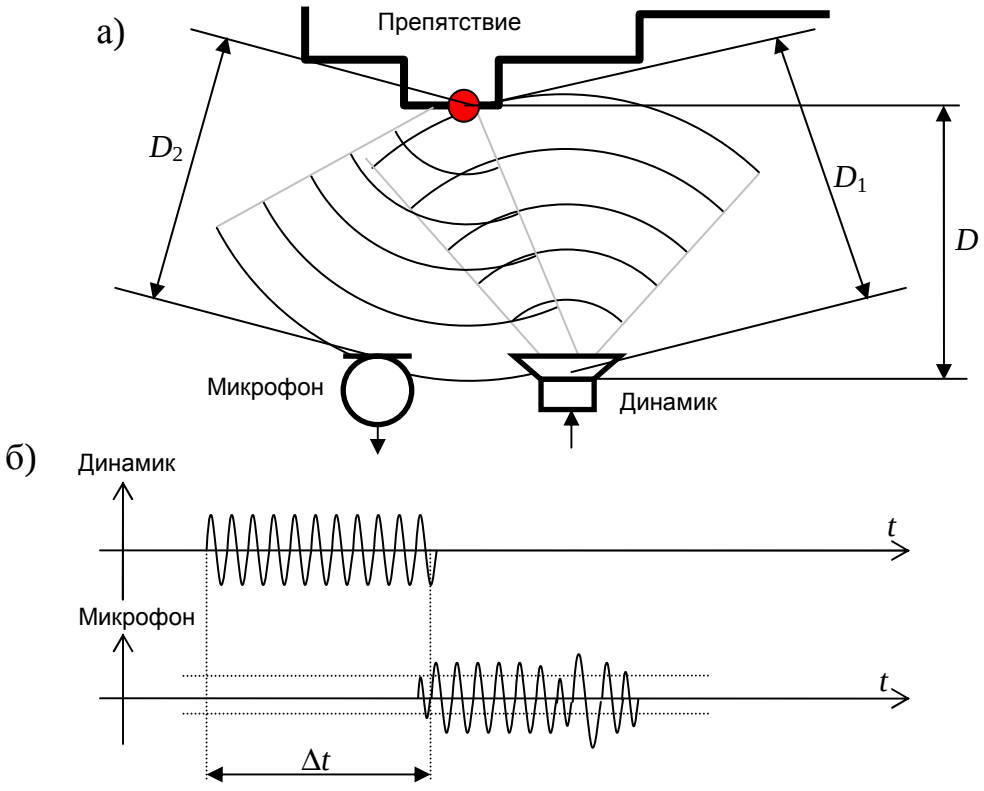

Рис. 10 Иллюстрация принципа работы ультразвукового дальномера: а) схема дальномера; б) графики излученного и принятого ультразвукового сигнала

Для формирования ультразвукового сигнала обычно используется пьезоэлектрический микрофон и динамик, т.к. обычные электромагнитные микрофоны и динамики не предназначены для данного звукового диапазона. Также в ультразвуковом дальномере имеется полосовой фильтр на данную звуковую частоту, чтобы отфильтровать чужеродные звуки.

Важной особенностью ультразвуковых дальномеров является наличие относительно широкого угла сканирования, обычно от 20 до 60 градусов. Дальномер формирует отметку дальности, если в зоне сканирования оказывается любое, даже небольшое препятствие.

Другой особенностью является ориентация поверхности препятствия относительно дальномера. Препятствие не регистрируется, если угол между осью дальномера и нормалью к поверхности более 70-80 градусов (в зависимости от гладкости поверхности), т.к. отраженная звуковая волна уходит в пустоту. При установке дальномера на робота следует понимать, что сигнал может отражаться от пола, если он попадает в конус сканирования дальномера.

Следует отметить, что конструктивно и микрофон и динамик устроены одинаково. Практически любой динамик может использоваться в качестве микрофона. Поэтому в некоторых дальномерах (например, в автомобильных парктрониках) микрофон и динамик совмещены в одно устройство: в момент излучения устройство работает в режиме динамика, а затем переключается в режим микрофона.

Примеры внешнего вида ультразвуковых дальномеров приведены на Рис. 11.

Рис. 11 Примеры внешнего вида ультразвуковых дальномеров

2.2.2. Интерфейсы подключения ультразвуковых дальномеров

Различные модели ультразвуковых дальномеров имеют различный интерфейс подключения. Известны четыре типа интерфейсов:

- RS-232;
- UART (производитель его называет RS-232 TTL);
- RS-485 (позволяет подключить несколько дальномеров в одному интерфейсу);
- ШИМ с раздельным каналом управления (Trigger Pulse, Echo Pulse)

- ШИМ с совмещенным каналом управления (Trigger/Echo по одному проводу).

Для подключения дальномеров по UART и по RS-485 управляющей ЭВМ требуется обеспечить специальный протокол. В Dyn-Soft RobSim 5 для реализации этого протокола имеется специальный блок, называемый «Блок управления дальномером».

При использовании 4-х проводного интерфейса ШИМ с раздельным каналом управления, управляющему устройству требуется обеспечить формирование сигнала TRIGGER (запуск измерения) и прием сигнала ECHO, формируемого дальномером. По длине ответного импульса ECHO управляющее устройство может определить расстояние до препятствия (Рис. 12). Уровень сигналов – TTL (обычно 5В в зависимости от напряжения питания). В паспорте дальномера указан коэффициент *K*, определяющий зависимость длины ответного импульса от расстояния *D* до препятствия. Обычно *K* равняется 58 мкс/см.

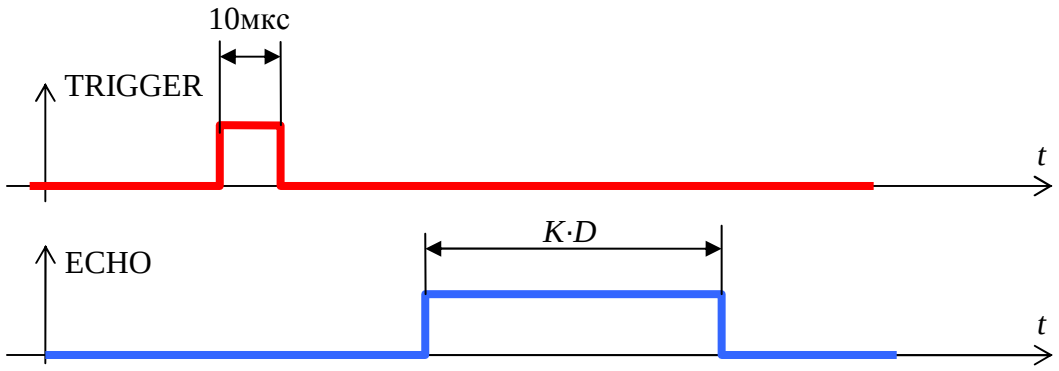

Рис. 12 Осциллограммы сигналов дальномеров интерфейсом ШИМ с раздельным каналом управления

При использовании 3-х проводного интерфейса ШИМ с совмещенным каналом управления управляющее устройство формирует на управляющей ножке дальномера импульс длительностью 10 мкс, затем переводит ножку в режим входа и дожидается на этой ножке формирования дальномером ответного импульса ECHO, длина которого определяет расстояние *D* до препятствия (Рис. 13).

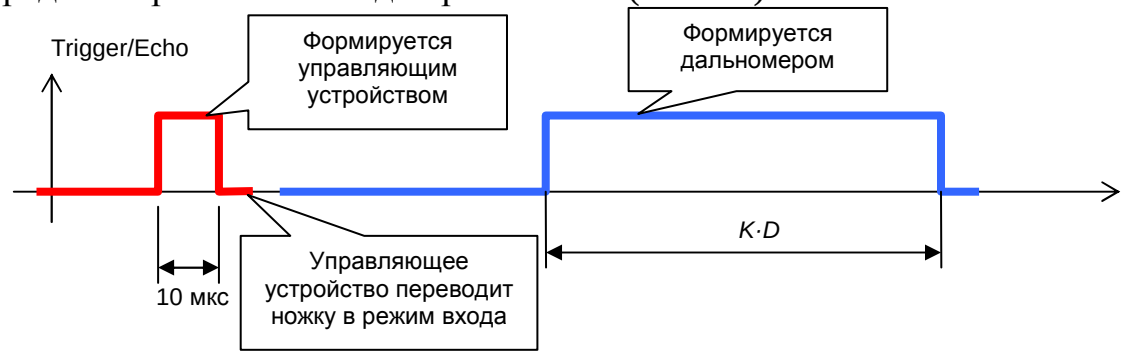

Рис. 13 Осциллограмма сигнала дальномера с интерфейсом ШИМ с совмещенным каналом управления

Уровень сигналов – TTL (обычно 5В в зависимости от напряжения питания). Коэффициент зависимости длины ответного импульса от расстояния также прописан в паспорте устройства и обычно равняется 58 мкс/см.

2.2.3. Использование ультразвуковых дальномеров в Dyn-Soft RobSim 5

Dyn-Soft RobSim 5 поддерживает несколько разновидностей ультразвуковых дальномеров. Для их установки необходимо в 3D Studio MAX установить на 3D-модель робота объект «Сенсор» и привязать его к объекту, к которому закреплен дальномер (Рис. 14). В параметрах объекта «Сенсор» следует выбрать тип «Дальномеры» и выбрать требуемую модель. Важно, что дальномер измеряет расстояние вдоль оси Z объекта «Сенсор».

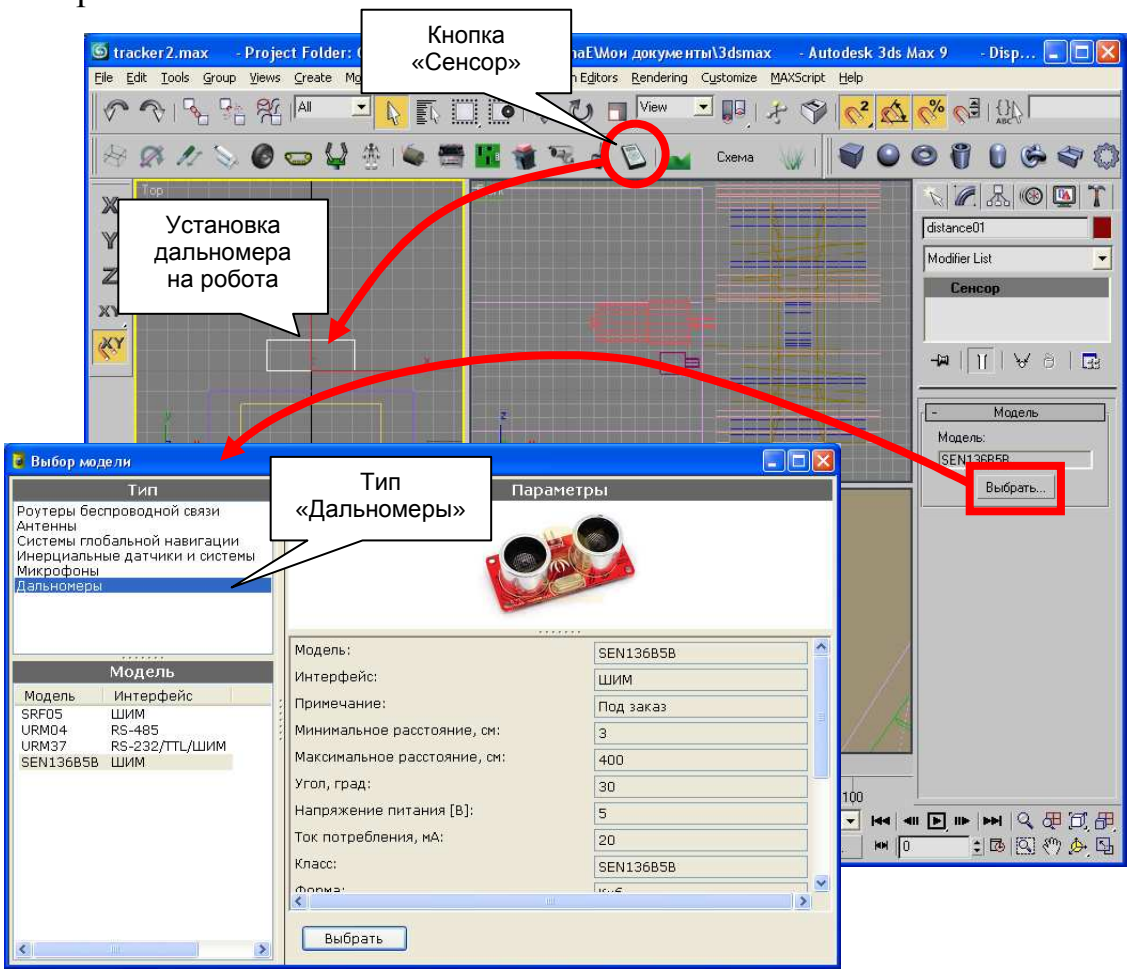

Рис. 14 Иллюстрация установки дальномеров в 3D Studio MAX

2.2.4. Подключение дальномеров с интерфейсом ШИМ с совмещенным каналом управления в Dyn-Soft RobSim 5

Для подключения ультразвуковых дальномеров с интерфейсом ШИМ с совмещенным каналом управления (Trigger/Echo по одному проводу, см. главу 2.2.2) следует использовать схему, изображенную на Рис. 15.

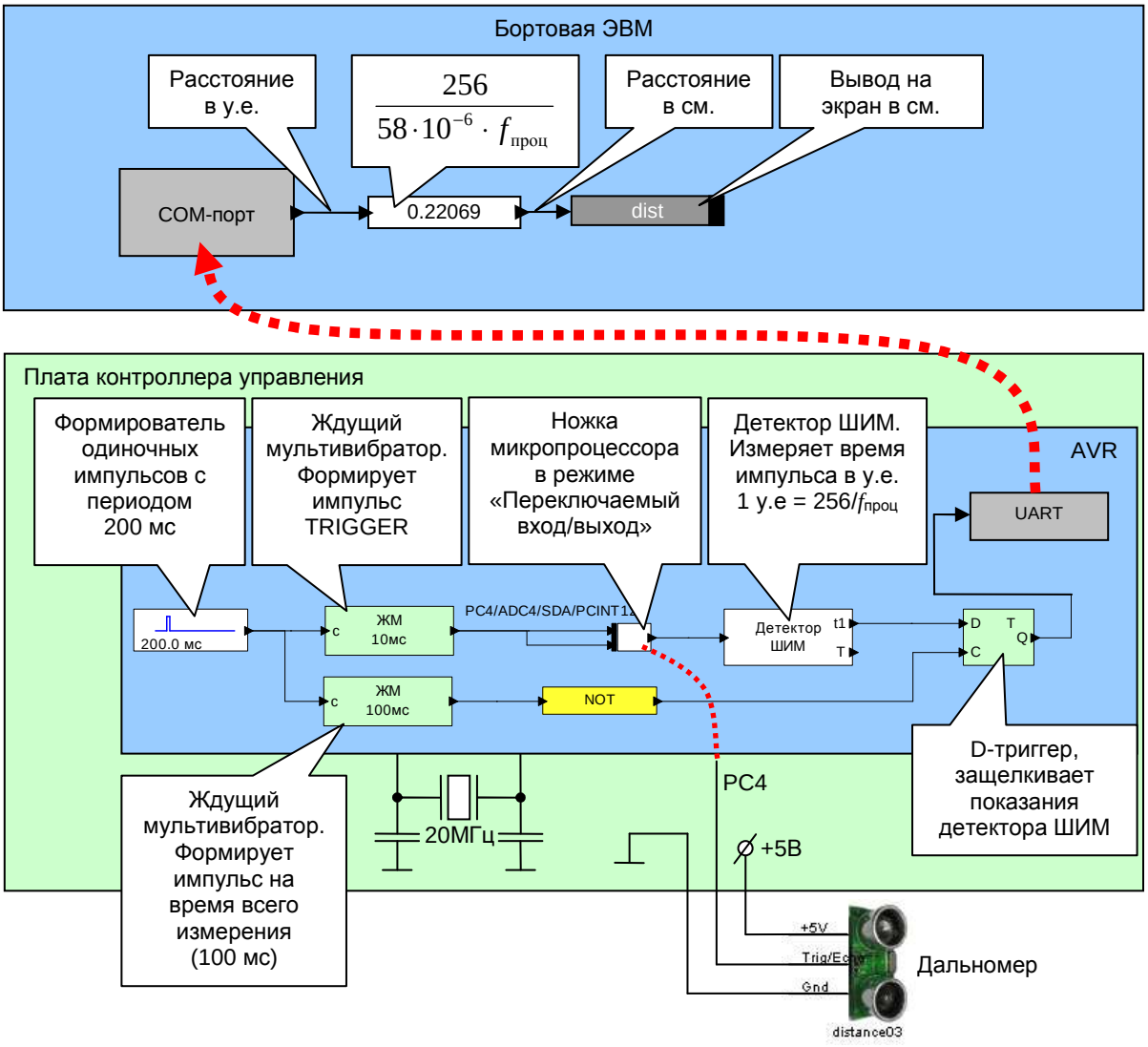

Рис. 15 Схема подключения дальномера с интерфейсом ШИМ с совмещенным каналом управления

На данной схеме управление дальномером осуществляется через вывод микропроцессора PC4. Данный вывод переведен в режим «Переключаемый вход/выход». В этом случае у входа появляется вход управления направлением работы порта (регистр DDR).

Для обеспечения работы измерения дальномера на структурной схеме программного обеспечения микроконтроллера установлен генератор одиночных импульсов, который раз в 200 мс формирует импульс на запуск

системы измерения. Запускающий импульс запускает два ждущих мультивибратора, заставляя на их выходах сформировать импульс стандартной длительности.

Первый ждущий мультивибратор формирует импульс длительностью 10 мс, который переводит вывод PC4 микропроцессора в режим выхода и формирует на нем логическую «1». Данный импульс запускает на дальномере процесс измерения дальности.

Спустя 10 мс первый ждущий мультивибратор вновь формирует на выходе «0», что заставляет микропроцессор вновь перевести вывод PC4 в режим входа.

Дальномер же начинает процесс измерения и, спустя некоторое короткое время, формирует на управляющем выходе импульс логической «1» длиной, пропорциональной расстоянию до препятствия. Этот импульс фиксируется выводом PC4 микропроцессора, сигнал с которого поступает на детектор ШИМ. Детектор ШИМ измеряет продолжительность единичного сигнала и формирует на своем выходе его длину в у.е. 1 у.е. равна 256 / $f_{\text{проц}}$ Где: $f_{\text{проц}}$ – частота микропроцессора в Гц. В данном случае $f_{\text{проц}}$ = 20 000 000 Гц.

Второй мультивибратор, запущенный совместно с первым, формирует импульс длиной 100 мс. Сигнал мультивибратора инвертируется элементом «НЕ». Таким образом, на входе «C» D-триггера в течение 100 мс формируется «0».

Спустя 100 мс второй мультивибратор вновь формирует «0», а инвертированный сигнал принимает значение «1». Передний фронт перехода сигнала из «0» в «1» на входе «C» D-триггера заставляет его «защелкнуть» значение на его входе «D» и держать это значение на своем выходе, пока не будет «защелкнуто» новое значение. Таким образом, триггер «защелкивает» результат измерения длины входного импульса.

Показания D-триггера передаются через UART из микропроцессора AVR на бортовую ЭВМ. Схема на Рис. 15 опускает подробности организации канала связи между бортовой ЭВМ и микропроцессором AVR.

Бортовая ЭВМ, приняв сигнал с измеренной длиной импульса, переводит его в сантиметры по формуле:

$$D\,[\text{см}] = d\,[\text{у.е.}] \cdot \frac{256\,[1/\text{у.е.}]}{58 \cdot 10^{-6}\,[\text{мкс/см}]\, f_{\text{проц}}[\text{Гц}]}$$

Где: d – длина импульса в у.е.; D – дальность до препятствия в сантиметрах; $58 \cdot 10^{-6}$ – это константа 58 мкс/см (см. главу 2.2.2).

2.2.5. Подключение дальномеров с интерфейсом ШИМ с раздельным каналом управления в Dyn-Soft RobSim 5

Схема подключения и принцип работы ультразвукового дальномера с интерфейсом ШИМ с раздельным каналом управления практически аналогичны дальномеру с совмещенным каналом управления (см. главу 2.2.4), за исключением того, что можно разнести сигналы TRIGGER и ECHO на разные выводы микропроцессора (Рис. 16). При таком подключении нет необходимости переводить управляющий вывод из одного режима в другой, поэтому структурная схема программного обеспечения микропроцессора упрощается.

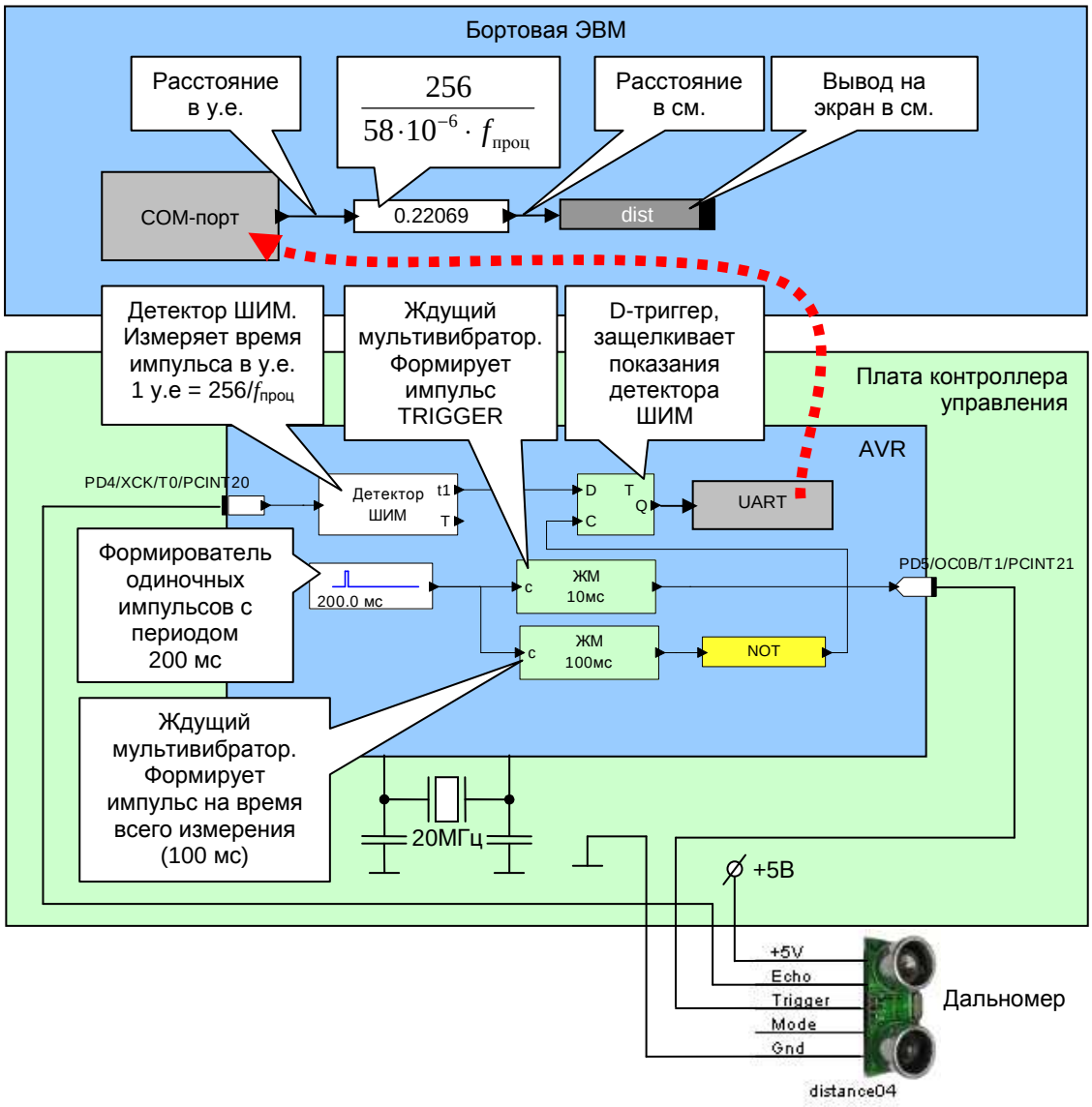

Рис. 16 Схема подключения дальномера с интерфейсом ШИМ с раздельным каналом управления

2.2.6. Подключение дальномеров с интерфейсом RS-232 в Dyn-Soft RobSim 5

Если дальномер имеет интерфейс RS-232, то его можно напрямую подключить к бортовой ЭВМ через COM-порт (Рис. 17). Следует обратить внимание о необходимости обеспечения питания дальномера. Для работы с дальномером в структуре программного обеспечения бортовой ЭВМ следует установить блок «Драйвер управления дальномером». В настройке COM-порта следует установить скорость передачи данных 9600, 8 бит данных, 1 стоповый бит без контроля четности.

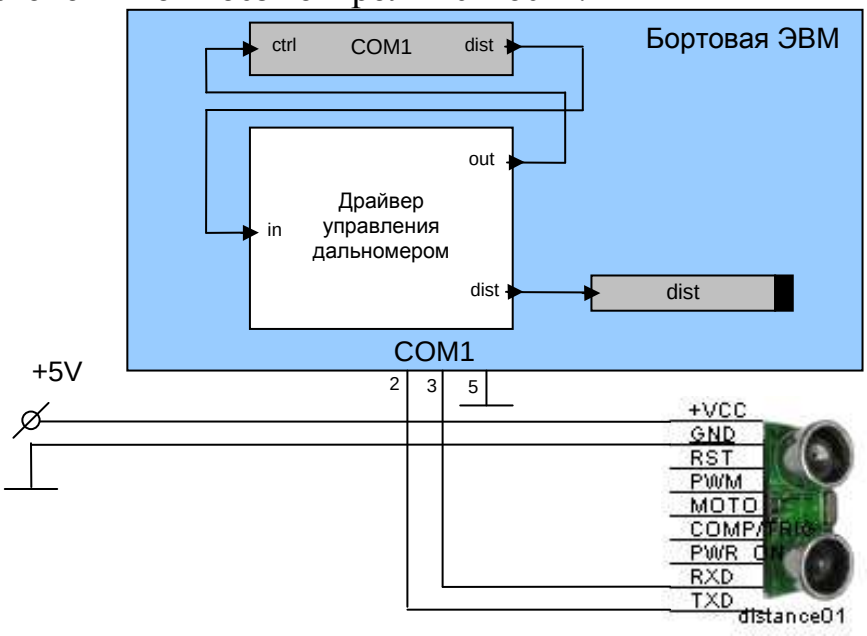

Рис. 17 Пример подключения дальномера с интерфейсом RS-232 к COM-порту бортовой ЭВМ

В качестве передаваемых данных следует установить сигнал по каналу 1 с типом данных «Управление дальномером», а в качестве принимаемых данных – установить сигнал по каналу 0 с типом данных «Ответ дальномера» (Рис. 18). Сигнал «ответ дальномера» с выхода COM-порта следует подать на вход «In» драйвера управления дальномером, а выход «out» драйвера управления дальномером следует подать на вход с типом данных «Управление дальномером» блока «COM-порт», согласно (Рис. 17). В этом случае на выходе «dist» будет формироваться дальность (типа short), измеряемая дальномером (в сантиметрах), или значение 0xFFFF в качестве признака отсутствия информации.

Если имеется такая необходимость (например, в случае отсутствия достаточного количества COM-портов у бортовой ЭВМ), то дальномеры с интерфейсом RS-232 можно подключить к микропроцессору AVR на плате контроллера управления. Но данный микропроцессор должен иметь 2 UART-порта (как, например, у микропроцессоров ATmega128 или

ATmega1284P) (Рис. 19). Настройка контроллера UART-порта должна быть такой же, как показано на Рис. 18.

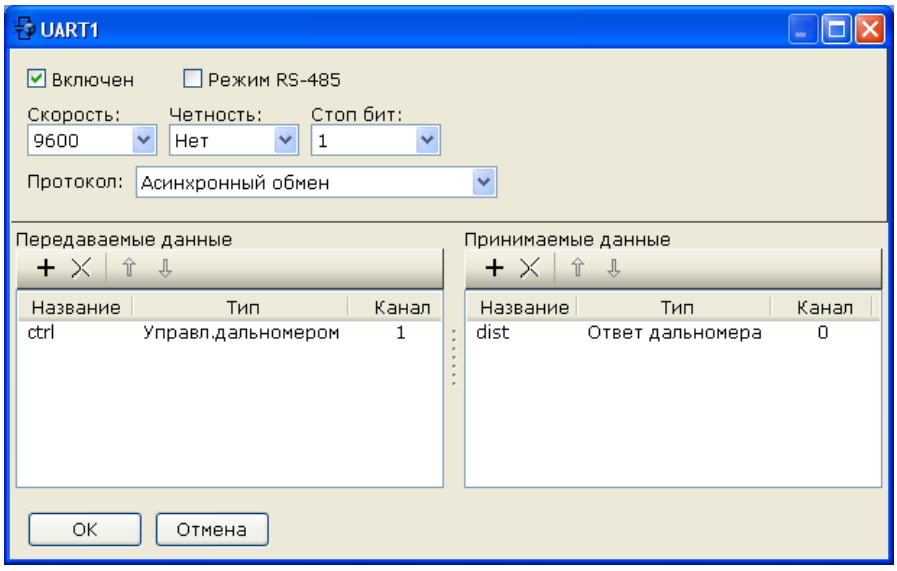

Рис. 18 Пример настройки COM-порта (UART-порта) для обеспечения связи с дальномером по RS-232

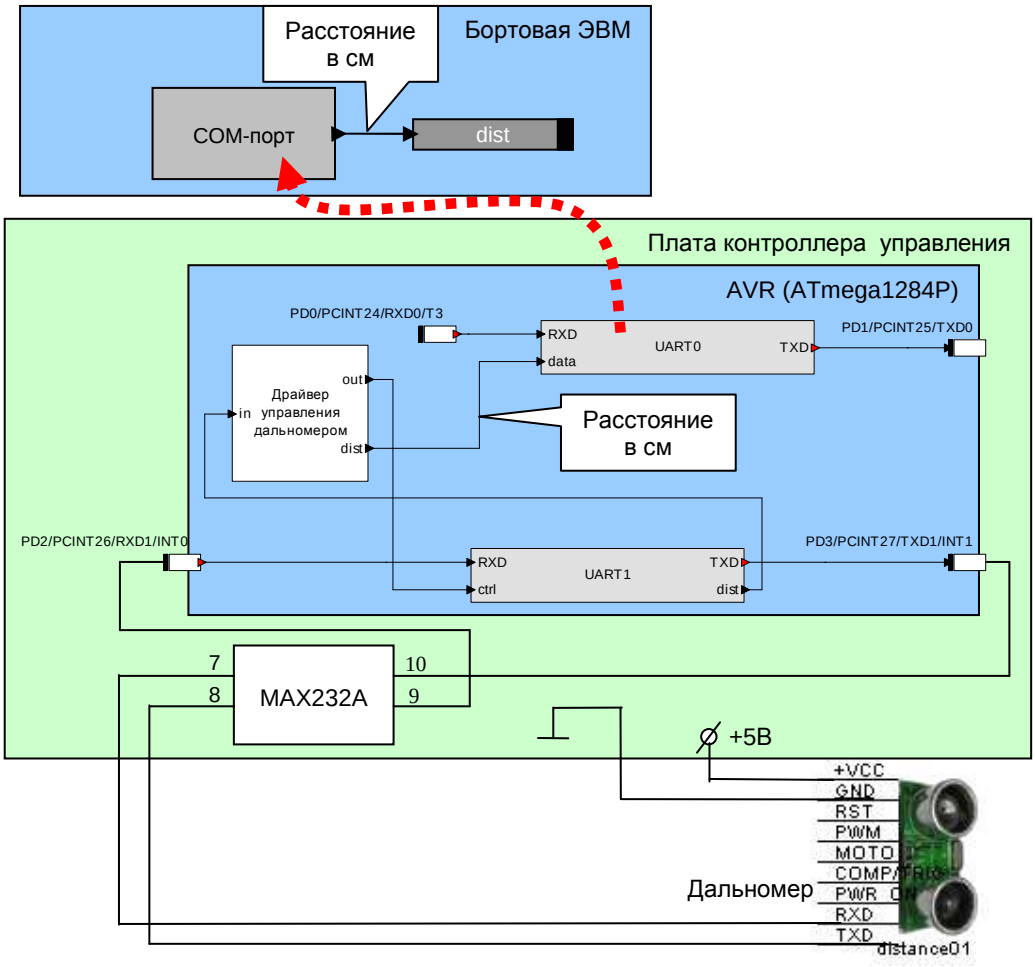

Рис. 19 Пример подключения дальномера с интерфейсом RS-232 к микропроцессору AVR

Следует отметить, что некоторые дальномеры, управляемые по RS-232, имеют встроенный датчик температуры. Организовать прием и выдачу данных о температуре можно в свойствах блока «Драйвер управления дальномером».

2.2.7. Подключение дальномеров с интерфейсом UART (TTL) в Dyn-Soft RobSim 5

Интерфейс UART поддерживается дальномером URM37. Для этого в самом дальномере имеются джамперы, переключающие режим с RS-232 на UART (производитель его называет RS-232 TTL). В Dyn-Soft RobSim 5 можно совершить данное переключение в свойствах блока «дальномер» в редакторе схем и подключений (Рис. 20).

Для подключения дальномера по интерфейсу UART к бортовой ЭВМ можно использовать схему из главы 2.2.6 (Рис. 17), при условии наличия у бортовой ЭВМ интерфейса UART (TTL-уровня 5V).

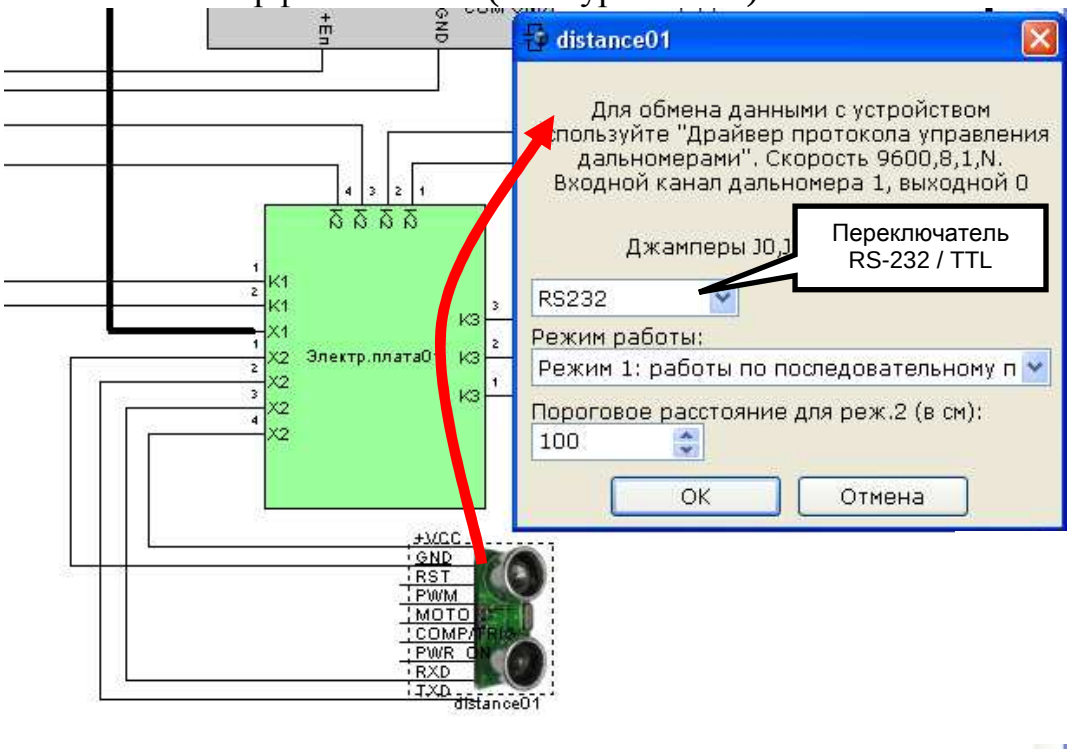

Рис. 20 Иллюстрация вызова свойств блока «Дальномер» в редакторе схем и подключений Dyn-Soft RobSim 5

Можно также подключить дальномер с интерфейсом UART к микропроцессору AVR. Для этого следует воспользоваться схемой из главы 2.2.6 (Рис. 19), с точностью до микросхемы MAX232. При подключении по UART нет необходимости в данной микросхеме, а дальномер может быть подключен непосредственно к UART-порту микропроцессора AVR.

2.2.8. Подключение дальномеров с интерфейсом RS-485 в Dyn-Soft RobSim 5

Интерфейс RS-485 позволяет к одной двухпроводной шине подключить сразу несколько дальномеров.

При использовании RS-485 для дальномеров следует обеспечить напряжение питания и провести дифференциальную пару от контроллера управления последовательно к каждому дальномеру (Рис. 21). Некоторые дальномеры (например, URM04) имеют продублированный набор выводов для удобства последовательного монтажа.

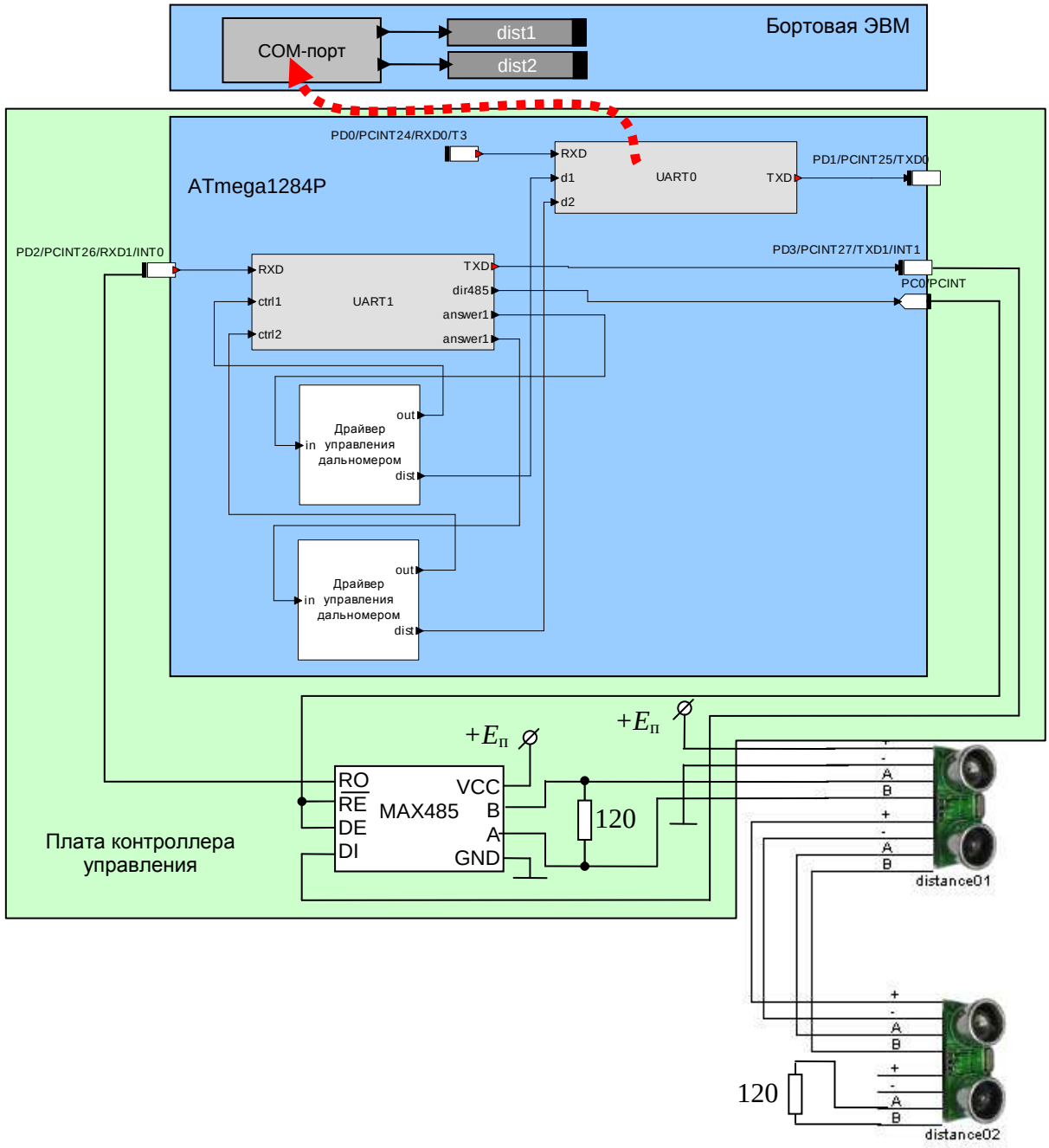

Рис. 21 Схема подключения дальномеров по интерфейсу RS-485 в Dyn-Soft RobSim 5

На обоих концах линии RS-485 рекомендуется установить резисторы по 100 Ом для организации токовой петли.

Каждый дальномер на линии RS-485 должен иметь собственный адрес. Данный адрес настраивается разработчиком по тому же интерфейсу RS-485 перед установкой дальномера на линию, тогда, подключен только один дальномер.

В Dyn-Soft RobSim 5 установка адреса (канала) дальномера производится в настройках свойств данного блока (Рис. 22)

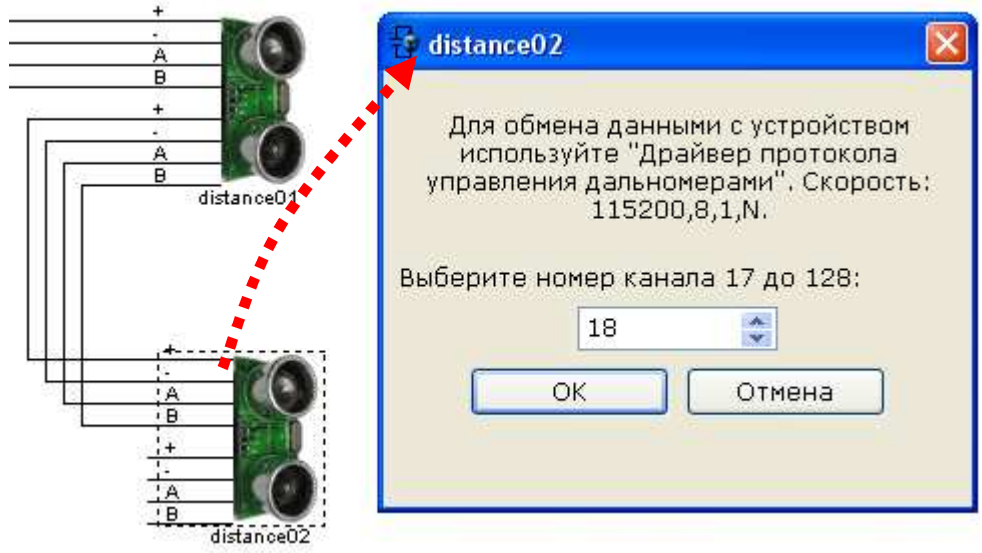

Рис. 22 Иллюстрация настройки свойств дальномера с интерфейсом RS-485

На плату контроллера управления следует установить драйвер интерфейса – микросхему MAX485, управление которой будет производиться с микропроцессора AVR. Важно использовать микропроцессор с двумя UART-контроллерами (например, ATmega128 или ATmega1284P). Один UART-контроллер будет обеспечивать связь с дальномерами, а другой – с бортовой ЭВМ.

В редакторе структуры программного обеспечения микропроцессора следует настроить UART-контроллер, обеспечивающий общение с дальномерами, следующим образом: установить скорость 115200, 8 бит данных, 1 стоповый бит, без контроля четности (Рис. 23). Включить с помощью соответствующей галочки поддержку RS-485. В этом случае у блока UART появляется дополнительный вывод управления направлением передачи данных по RS-485. В качестве передаваемых данных следует добавить несколько сигналов типа «Управление дальномером» (по числу дальномеров) по каналам, соответствующие каналам, заданным в свойствах этих дальномеров. В качестве принимаемых данных следует добавить сигналы типа «Ответ дальномера» (по числу дальномеров) по тем же каналам, что и передаваемые данные.

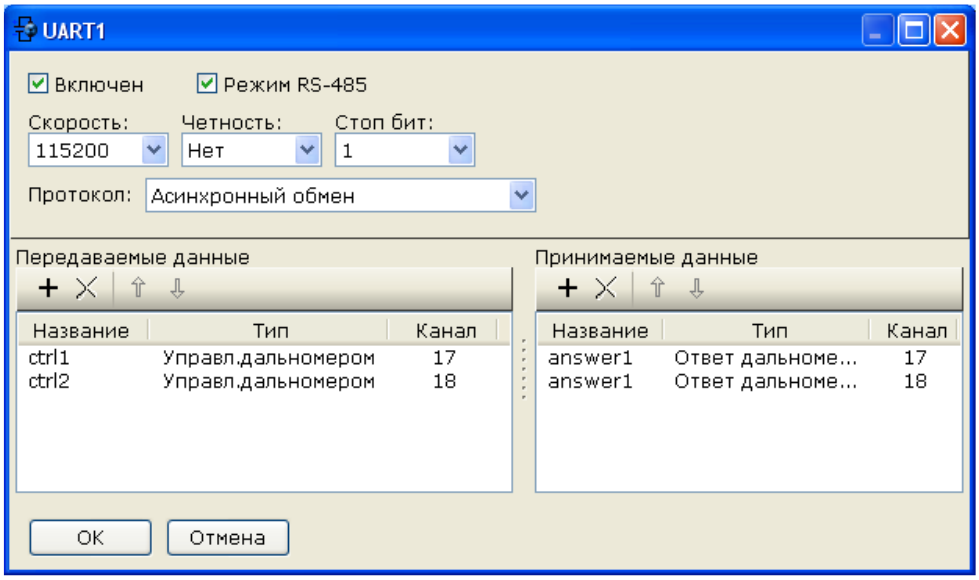

Рис. 23 Пример настройки свойств UART-контроллера для обеспечения связи с дальномерами по интерфейсу RS-485

Выход «dir» блока контроллера UART следует вывести через любой свободный вывод микропроцессора и подключить его на управление направлением передачи данных RE и DE на микросхеме MAX485. Следует отметить, что сигнал RE (Read Enable) инверсный, а DE (Data Enable) – прямой. Поэтому сигнал логической «1», подключенный одновременно к данным ножкам, отключает прием (RE), и включает передачу (DE). И наоборот.

Для управления дальномерами в структуре программного обеспечения микропроцессора следует предусмотреть несколько блоков «Драйвер управления дальномером» (по числу дальномеров) и подключить их к блоку контроллера UART (Рис. 21).

На каждом выходе «dist» драйверов управления дальномерами будет формироваться измеряемая дальность в сантиметрах (тип short) или 0xFFFF, если дальномер не обнаруживает перед собой препятствия. Разработчик может передать ее на бортовую ЭВМ для отображения на экране и использования по назначению.

2.3.Прочие информационные сенсоры

Кроме уже перечисленных камер и ультразвуковых дальномеров в системах очувствления роботов используют прочие датчики, применяющиеся для специальных нужд. Среди них:

- Лазерные дальномеры.
- Сканирующие лазерные дальномеры (Лидары).
- Датчики освещенности.
- Датчики температуры и давления.

- Тензодатчики (реагирующие на прикосновение).
- Приборы ночного видения (оснащенные ЭОП).
- Инфракрасные камеры.
- Тепловизоры.
- Инклиномеры (датчики наклона).

Данные датчики применяются для выполнения роботом специальных задач. Их применение не носит массовый характер, причем некоторые из перечисленных типов датчиков серийно не выпускаются. На некоторые сенсоры (например, на ЭОП) имеются ограничения экспорта и импорта в Россию.

Интерфейс подключения перечисленных сенсоров, как готовых изделий, может сильно отличаться даже в рамках одного типа датчиков. Так, например, тепловизоры могут быть как с аналоговым выходом (PAL), так и с USB, и PCI-E, и LAN (Ethernet).

Поэтому в Dyn-Soft RobSim 5 перечисленные типы датчиков не представлены.

3. Информационно-измерительная система интеллектуальных мобильных роботов

3.1.Стереозрение

3.1.1. Принцип работы стереозрения

Стереозрение или бинокулярное зрение – это двухкамерная система технического зрения, обеспечивающая возможность определения дальности до объектов в поле зрения двух камер.

Эффект стереозрения обеспечиваться за счет *параллакса*.

Параллакс – расстояние в пикселях между одной и той же точкой объекта, зафиксированной на двух изображениях, снятых в разных точках пространства в один и тот же момент времени (Рис. 24).

Чем ближе расстояние до объекта, тем больше его параллакс. По длине параллакса Δx_i можно определить расстояние Z_i до i-ой точки объекта по формуле:

$$Z_i = \frac{D \cdot F_x}{\Delta x_i}$$

Где: D – расстояние между камерами (база); F_x – масштабный коэффициент:

$$F_x = \frac{W / 2}{\mathrm{tg}\left(\frac{FOV_x}{2}\right)}$$

Здесь: W – ширина кадра в пикселях; FOV_x – угол угла обзора камеры по горизонтали.

Данная зависимость справедлива, если оптические оси камер строго (с точностью до 1 пикселя на бесконечности) параллельны. Однако на практике этого очень сложно добиться.

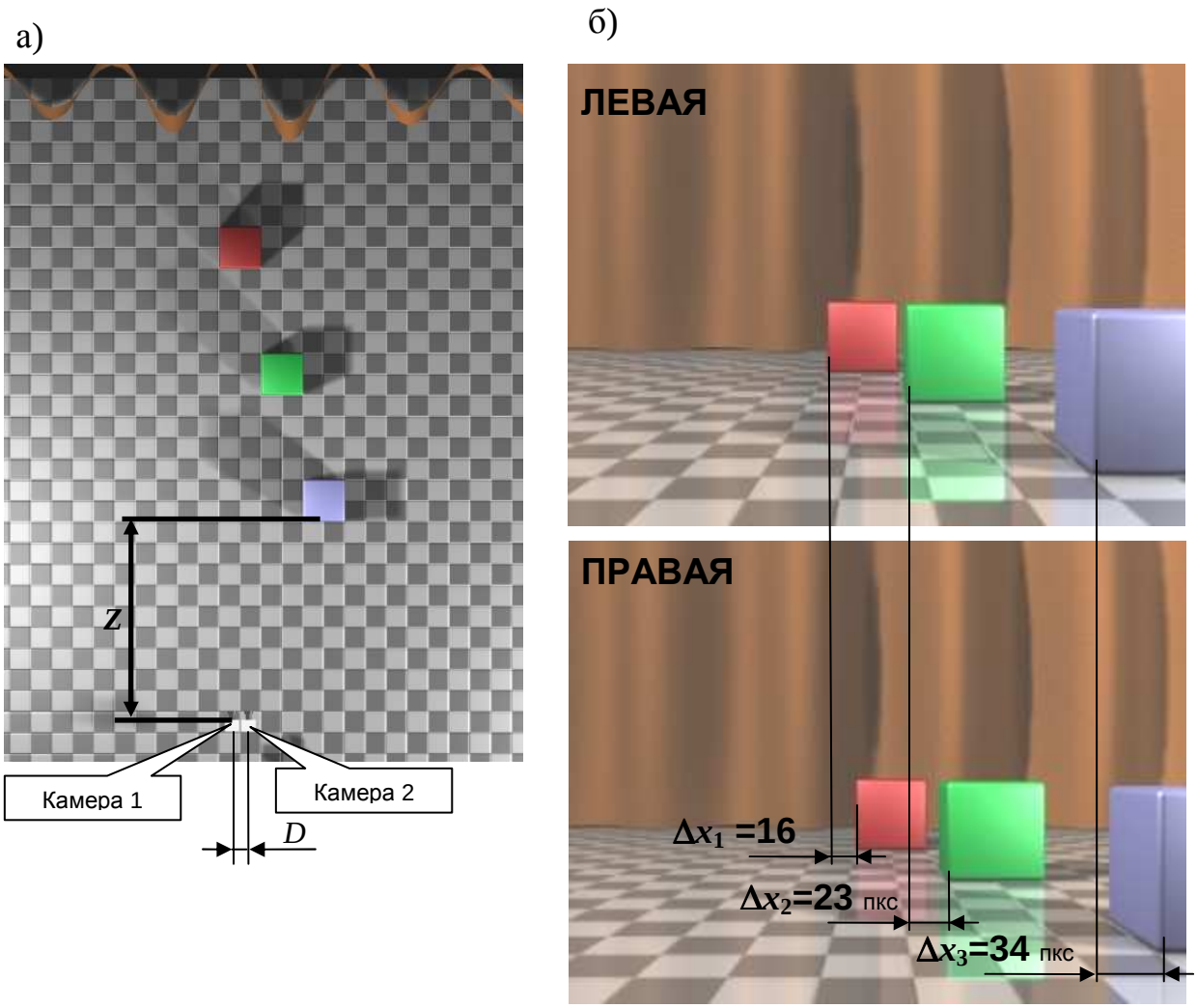

Рис. 24 Иллюстрация принципа стереозрения: а) сцена, вид сверху; б) изображение с левой и правой камеры с обозначенной длинной параллакса Δx_i

Поэтому для калибровки используют следующую схему (Рис. 25): путем сдвига изображений Δx_0 совмещают изображение с двух камер так, чтобы на двух изображениях совпала какая-либо точка с известной дальностью *B*. На Рис. 25 это точка *A*. Длины параллаксов Δx_i после такого сдвига становятся меньше. Для определения расстояния Z_i до объекта после калибровки камер используют формулу:

$$Z_i \cong B \cdot \frac{\Delta x_0}{\Delta x_i + \Delta x_0}$$

Следует обратить внимание, что после калибровки теряется зависимость как от угла обзора камер, так и от расстояния между камерами.

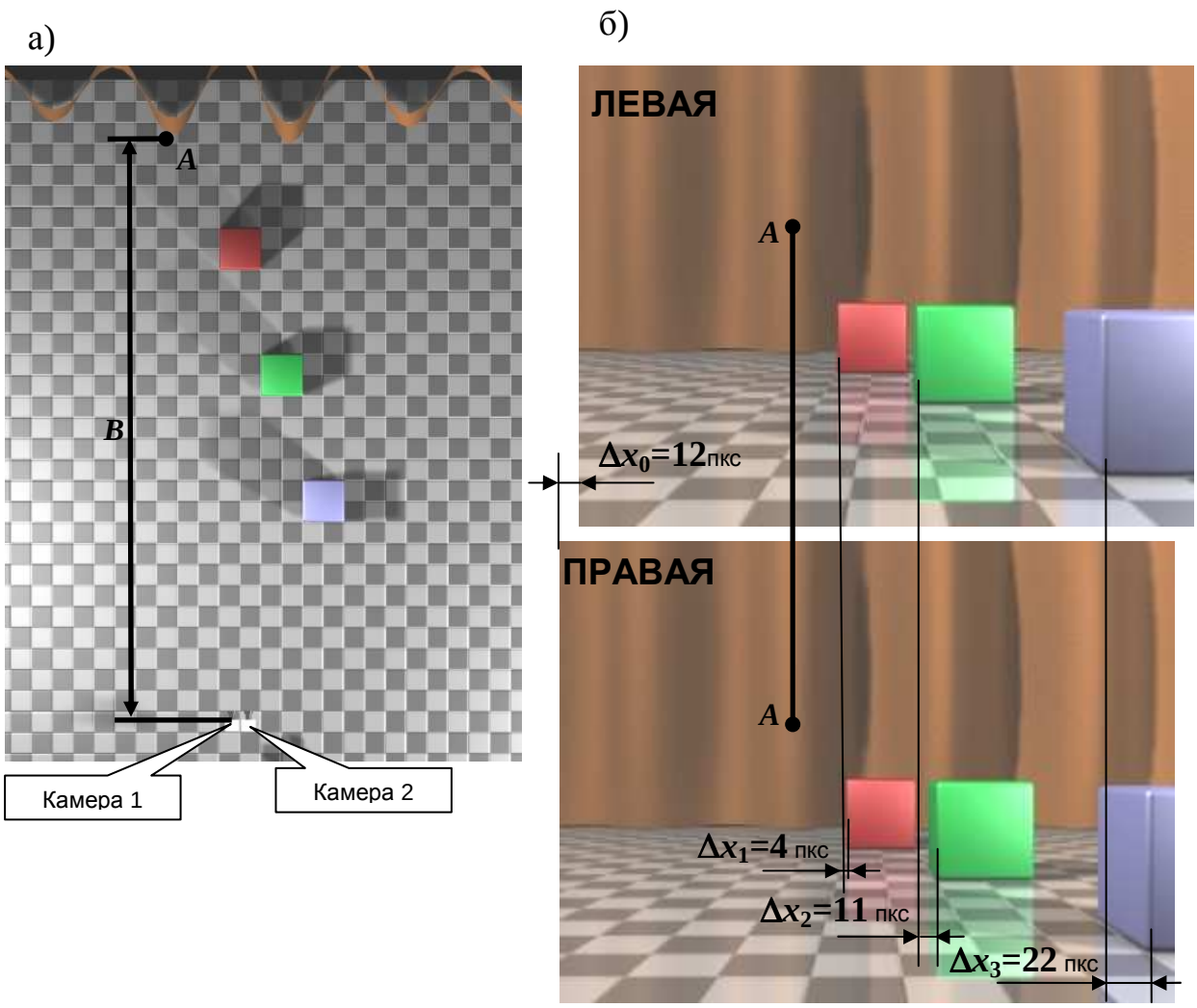

Рис. 25 Иллюстрация принципа калибровки камер системы стереозрения: а) сцена, вид сверху; б) изображение с левой и правой камеры

Координаты X_i, и Y_i точки легко найти по формулам:

$$X_i = \left(x_i - \frac{W}{2} \right) \cdot \frac{Z_i}{F_x}$$

$$Y_i = \left(\frac{H}{2} - y_i \right) \cdot \frac{Z_i}{F_x}$$

Где: x_i, y_i – экранные координаты точки (в пикселях); W, H – ширина и высота изображения (в пикселях); F_x – масштабный коэффициент.

Следует отметить, что экранная ось y направлена вниз, в то время, как мировая ось Y – вверх. Поэтому знак по оси y отличается от знака по оси x.

Для автоматического поиска одинаковых точек на двух изображениях используют корреляционные методы. Причем при таком поиске важно искать совпадение не отдельного пикселя, а группы пикселей, т.к. цвет одного отдельного пикселя может совпасть совершенно

случайно. Поиск можно производить только контрастных точек объекта, т.к. в случае отсутствия контраста будет фиксироваться совпадение при практически любом сдвиге изображения (как, например, для точки центра грани кубика на рисунке).

3.1.2. Особенности системы стереозрения

Контраст

Алгоритмы стереозрения способны работать только при условии наличия у объектов сцены большого количества контрастных точек. Без наличия контраста невозможно определить одну и ту же точку объекта на двух изображениях, а, стало быть, невозможно определить ее параллакс. Несмотря на то, что имеются алгоритмы достаивающие изображение в этих точках, артефакты неизбежны.

Объекты типа «Горизонтальные трубы»

Алгоритмы стереозрения не способны определить расстояние до объектов типа «горизонтальные трубы», несмотря на их высокий контраст. Дело в том, что горизонтальные трубы, также как и неконтрастное изображение, совмещаются друг с другом при любом горизонтальном смещении изображений.

Синхронность

Для обеспечения функциональности алгоритмов стереозрения необходимо чтобы изображение с двух камер было снято в один и тот же момент времени, или за время между кадрами, снятыми с одной и другой камерами, сцена не сдвинулась ни на пиксель (в проекции на камеры). В противном случае параллаксы становятся зависимыми в большей степени от скорости движения объектов (или скорости движения камер относительно объектов), чем от расстояния до них.

Для обеспечения работоспособности алгоритмов стереозрения во время движения, используются механизмы синхронизации камер. Если это сделать невозможно, то роботу приходится всякий раз останавливаться, чтобы осмотреться.

Точность

Длина параллакса измеряется в целых пикселях, причем зависимость гиперболическая. Поэтому чем больше диапазон значений фиксируемых параллаксов, тем точнее можно определить расстояние до объекта. Так, например, если параллакс дальнего объекта 1 пиксель, а ближнего – 3 пикселя, то между этими объектами всего 3 дискреты расстояния причем с неравномерным шагом.

Повысить точность алгоритмов стереозрения можно путем увеличения диапазон значений параллаксов. Это можно сделать двумя способами:

1. Увеличить расстояние между камерами (увеличить базу).
2. Увеличить разрешение камер.

При увеличении расстояния между камерами существенно увеличиваются параллаксы. Однако бесконечно увеличивать базу нельзя, т.к. в этом случае, во-первых, близкие объекты перестанут попадать в поле зрения обоих камер, а, во-вторых, на изображениях с двух камер будут иметься настолько большие различия, что автоматически определить одну и ту же точку объекта на этих двух изображениях будет невозможно.

Поэтому нормальное расстояние между камерами для определения расстояний до объектов, среднее удаление которых составляет 50-100 см, составляет 3-7 см.

При увеличении разрешения камер параллаксы увеличиваются из-за того, что одним и тем же угловым размерам соответствует большее число пикселей. Однако при повышении разрешения существенно возрастает вычислительная нагрузка на процессор.

3.1.3. Использование системы стереозрения в Dyn-Soft RobSim 5

Для использования стереозрения в Dyn-Soft RobSim 5 требуется подключить к роботу две камеры. Про подключение камер подробно описано в главе 2.1.4.

Для реализации системы стереозрения в Dyn-Soft RobSim 5 имеется специальный блок «Построитель карты видимой области по стереозрению», расположенный в редакторе программного обеспечения бортовой ЭВМ на закладке «Интеллект» (Рис. 26).

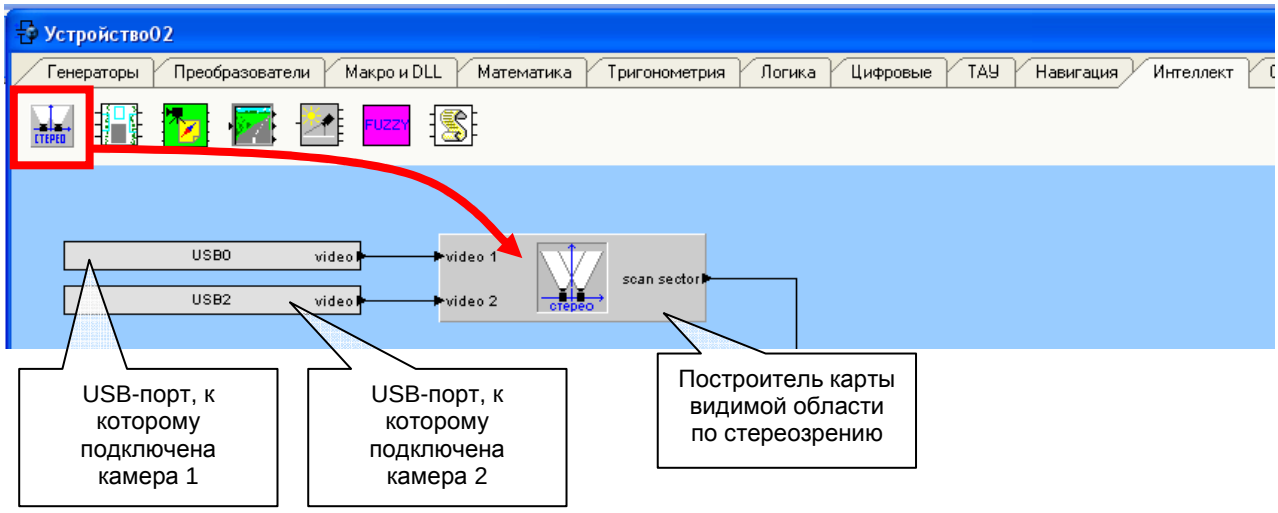

Рис. 26 Подключение построителя карты видимой области по стереозрению в Dyn-Soft RobSim 5

Данный блок позволяет по изображению с двух камер построить, так называемый, *сектор сканирования*.

Сектор сканирования представляет собой набор из одной или нескольких двухмерных карт расположения препятствий в видимом секторе в каждом срезе по высоте (Рис. 27).

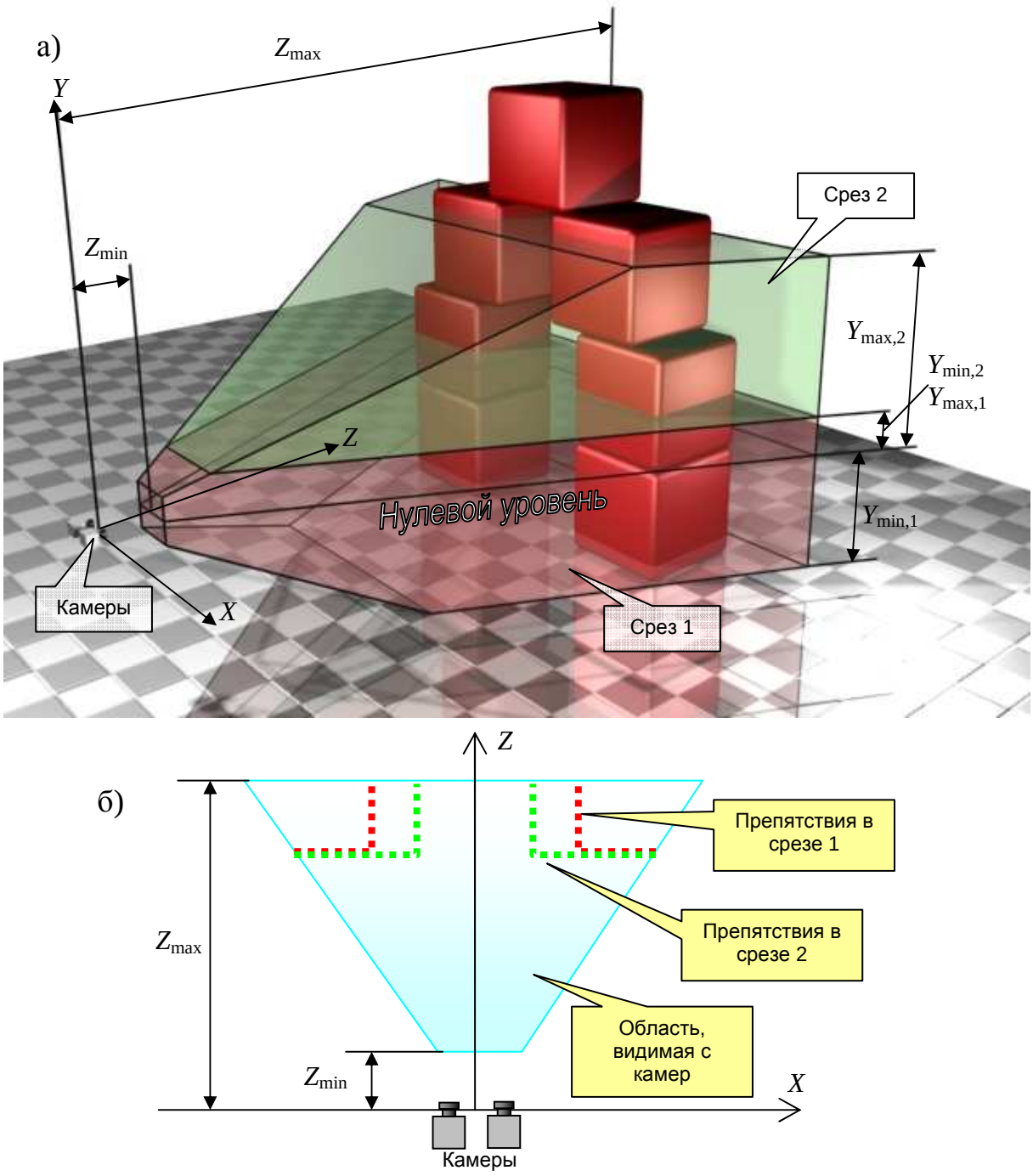

Рис. 27 Иллюстрация понятия «сектор сканирования»: а) расположение срезов в пространстве; б) отображение срезов на карте видимой области

Для построения сектора сканирования сначала по стереозрению определяются трехмерные координаты точек препятствий в стереокадре видеоизображения. Для этого используется система координат, в которой ось Y направлена вверх, ось X – направо от оптической оси камер, а ось Z – вдоль оптической оси камер. За начало координат принимается точка между камерами. Координаты точек вдоль оси высоты Y разделяются на заранее определенные срезы. Количество срезов определяется пользователем. Обычно используется только 1 срез, но в некоторых случаях необходимо использовать несколько. У каждого i-ого среза задан диапазон высот от $Y_{min,i}$ до $Y_{max,i}$. Из точек, попадающих в срез, строится двухмерная карта данного среза в плоскости X, Z. Таким образом, формируются несколько карт по разным высотам.

Во избежание артефактов задается общий для всех срезов диапазон дальностей от Z_{min} до Z_{max}. Точки, непопадающие в один из срезов или в диапазон дальностей, не используются в построении.

Разделение сектора сканирования на срезы может быть необходимо для тех роботов, у которых различная ширина на разных высотах. Например, широкое шасси, но узкий манипулятор. Поэтому, препятствия, которые могут мешать шасси, не всегда могут мешать манипулятору.

На Рис. 28 показан внешний вид диалогового окна настройки блока «Построитель карты видимой области по стереозрению». В данном диалоговом окне пользователю предлагается задать диапазон дальностей (Z_{min}, Z_{max}), расстояние между камерами D, угол обзора камер, а также определиться с количеством срезов и диапазоне высот каждого среда ($Y_{min,i}$, $Y_{max,i}$).

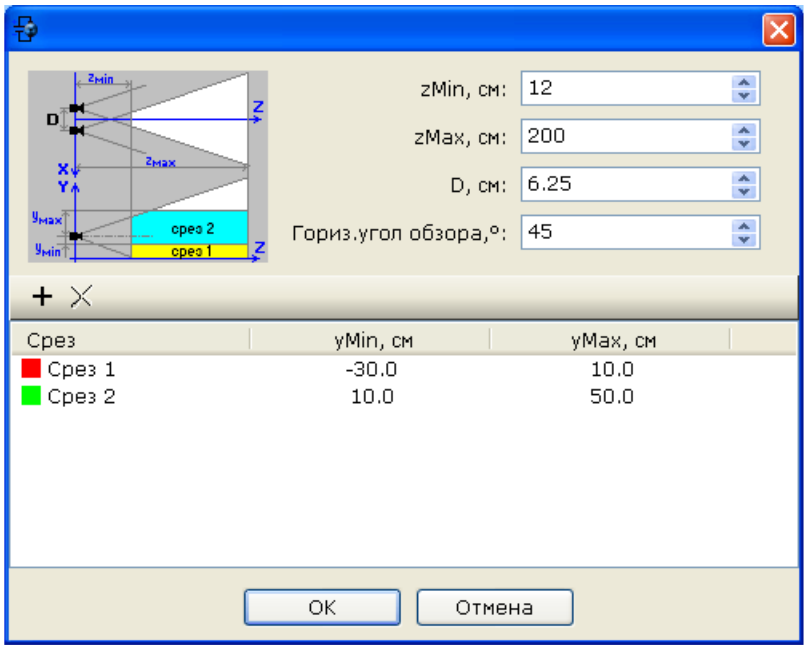

Рис. 28 Внешний вид диалогового окна настройки блока «Построитель карты видимой области по стереозрению

Для адекватного построения сектора сканирования пользователь должен точно задать расстояние между камерами (*D*) и угол обзора камер, в противном случае могут возникать нелинейные искажения формы препятствий на карте.

На выходе блока «Построитель карты видимой области по стереозрению» формируется сигнал с типом данных «сектор сканирования». Данный тип сигнала совместим с сигналом типа «Видео» и может быть отрисован блоком «Видео» на пульте оператора.

Разработчику робота, желающему использовать стереозрение в Dyn-Soft RobSim 5, настоятельно рекомендуется в целях отладки передать сигнал «сектор сканирования» по WiFi на пульт оператора и вывести на экран с помощью блока «Видео». При последующей отладки следует уделить внимание ширине сектора сканирования, ровности построения линий из элементов препятствий и формированию сектора сканирования в движении.

3.1.4. Рекомендации по установки системы стереозрения на мобильного робота

При установке системы стереозрения на мобильного робота следует обратить внимание на место ее установки. Естественное желание пользователя установить ее на место человеческих глаз – неоправданно. Дело в том, что чем выше будет установлены камеры, тем меньше в них будет попадать пол в ближней зоне перед роботом. Данная ближняя зона представляет больший интерес, чем дальняя (Рис. 29).

Поэтому система технического зрения должна быть установлена невысоко.

Но и совсем низко не стоит устанавливать камеры, т.к. в этом случае снижается обзор, теряется контроль над верхней частью ближней зоны, а объективы камер, установленных низко над землей, быстрее загрязняются.

В Dyn-Soft RobSim 5 поддерживаются только системы стереозрения, оптические оси которых направлены вдоль оси горизонта. Если данное условие не соблюдается, то сектор сканирования будет формироваться с искажениями. При небольших наклонах эти искажения будут несущественными, однако при углах наклона 20-30º искажения начнут резко проявляться.

Поэтому в Dyn-Soft RobSim 5 вообще не рекомендуется использовать наклон системы технического зрения.

Для расширения поля обзора камер можно устанавливать камеры на поворотную платформу. В этом случае перед началом поворота на месте робот может просканировать пространство в направлении поворота и оценить необходимость такого поворота. Но для использования данного механизма разработчику следует вносить дополнительные алгоритмы в систему управления поведением робота.

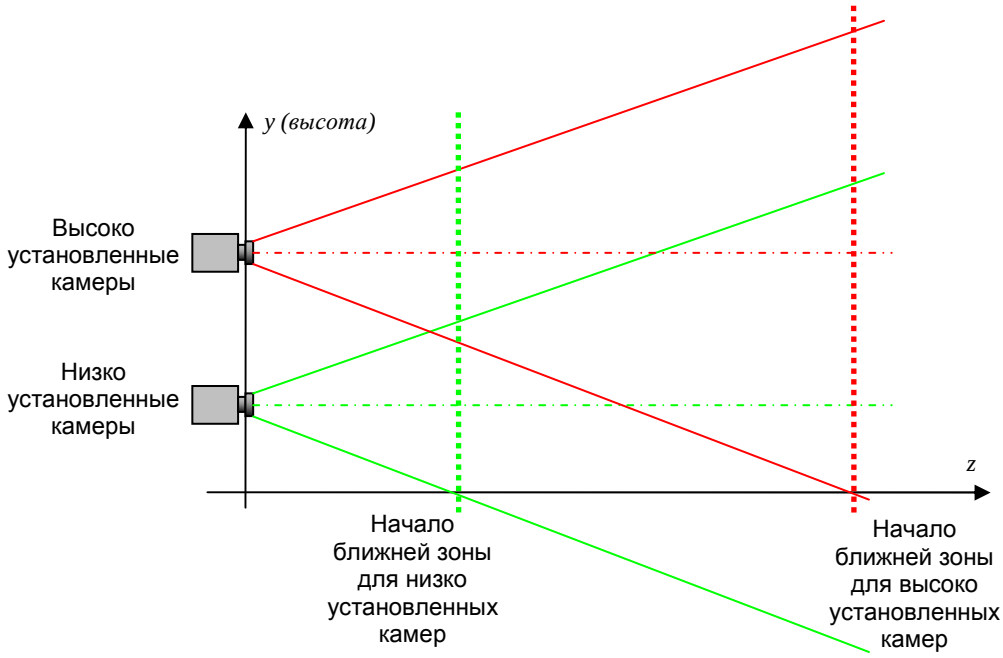

Рис. 29 Иллюстрация начала ближней зоны для высоко и низко установленных камер

3.2.Локальная карта местности

3.2.1. Понятие локальной карты местности

При использовании мобильным роботом системы стереозрения или дальномеров возникает проблема, связанная с ограниченным углом обзора этих систем. Так, препятствия, которые могут быть обнаружены непосредственно с помощью этих систем, обычно находятся в дальней зоне и не представляют для робота особой опасности. А наибольшую опасность представляют препятствия, которые находятся в ближней зоне, не просматриваемой с помощью сенсора. Однако нельзя сказать, что препятствия в ближней зоне никогда не обнаруживались сенсорами. Эти препятствия когда-то находились в дальней зоне, но по мере движения робота вышли за границы ближней видимой зоны обнаружения (Рис. 30).

Робот, изображенный на рисунке, следуя логике поведения, основанной только на базе системы очувствления, ошибочно повернет вправо, после чего произойдет столкновение с препятствиями. Важно отметить, что показанная на рисунке ситуация намного чаще встречается в реальной жизни, чем та идеальная, когда опасные препятствия находятся в зоне обнаружения систем очувствления робота.

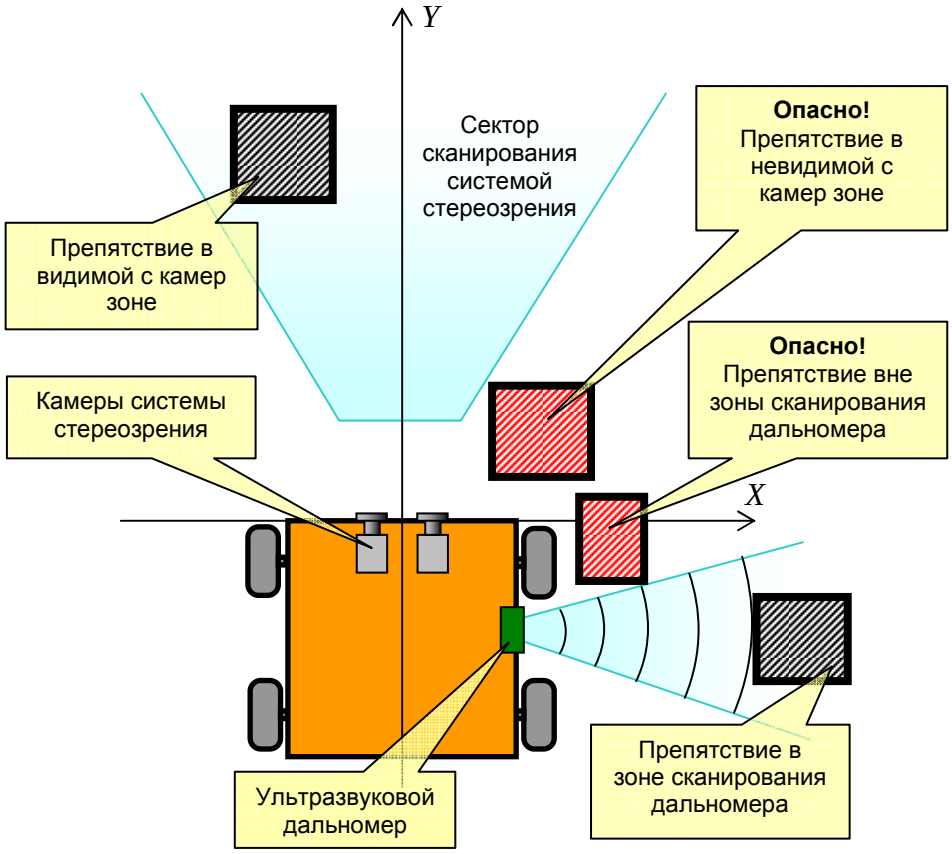

Рис. 30 Препятствия, представляющие и не представляющие опасности для робота

Решением проблемы может стать формирование карты препятствий вокруг робота. Несложно заметить, что по мере движения робота опасные препятствия, обозначенные на рисунке, все же однажды фиксировались системой очувствления робота. Если бы в момент обнаружения эти препятствия попали бы на карту и передвигались по карте по мере движения робота, то данной проблемы не возникло.

Поэтому для функционирования интеллектуального мобильного робота вводится понятие *локальная карта местности*.

Локальная карта местности представляет собой карту расположения отметок от препятствий (т.е. точек с координатами X_i, Y_i) вокруг робота. Центром локальной карты местности всегда является центр вращения робота.

При поступлении информации с систем очувствления робота с локальной карты местности удаляются отметки от препятствий, находящихся в зоне обнаружения данного сенсора, а затем эта зона заполняется новой информацией от данной системы очувствления. От системы технического зрения на локальной карте местности формируется карта препятствий на основании сектора сканирования, а от дальномеров формируется дуга из отметок от препятствий с радиусом, совпадающим с

показаниями дальномера, заполняющая весь угол сканирования дальномера.

Затем отметки от препятствий перемещаются по карте местности, используя информацию от системы навигации робота. По мере движения данные точки выходят за пределы видимых с сенсоров зон, но остаются на карте. Обычно системы навигации, основанной на датчиках обратной связи на колесах (см. главу 4.3.3), вполне достаточно для адекватной работы алгоритмов формирования локальной карты местности.

Перемещение элементов карты производится по формулам:

$$X_i^{<new>} = X_i \cdot \cos\Delta\alpha - Y_i \cdot \sin\Delta\alpha$$

$$Y_i^{<new>} = X_i \cdot \sin\Delta\alpha + Y_i \cdot \cos\Delta\alpha - \Delta L$$

Где: $X_i^{<new>}$ и $Y_i^{<new>}$ – новые координаты точки с координатами X_i, Y_i; $\Delta\alpha$, ΔL – угол поворота робота и линейное перемещение робота за такт расчета. Если на входе системы локальной карты местности угловая скорость поворота ω и линейная скорость движения v робота, то:

$$\Delta\alpha = \omega \cdot \Delta t$$

$$\Delta L = v \cdot \Delta t$$

Здесь Δt –такта расчета.

Локальная карта местности имеет ограниченный размер, обычно 1-2 метра вперед, 1 метр в стороны и 0.5 метра назад. Отметки от препятствий, которые по мере движения робота выходят за пределы локальной карты местности, удаляются. Если этого не делать, то на локальной карте местности могут возникать ложные отметки от далеких препятствий, которые давно попали на карту, но из-за погрешности работы системы навигации накопили ошибку интегрирования и оказались совершенно не в правильных координатах.

Иллюстрация работы алгоритмов по формированию локальной карты местности показана на Рис. 31. Из рисунка видно, что локальная карта местности на пульте оператора состоит из препятствий в видимой зоне и препятствий в невидимой зоне, попавших в нее по мере приближения робота к стене.

На локальной карте местности можно отмечать препятствия по нескольким срезам высоты (см. главу 3.1.3), обеспечивая проход робота по карте по разным высотам.

Процесс формирования карты местности схож с процессами формирования карты в мозгу человека. Ведь когда человек идет, он обычно специально не смотрит под ноги, но прекрасно осознает все препятствия на пути, потому что видел их однажды.

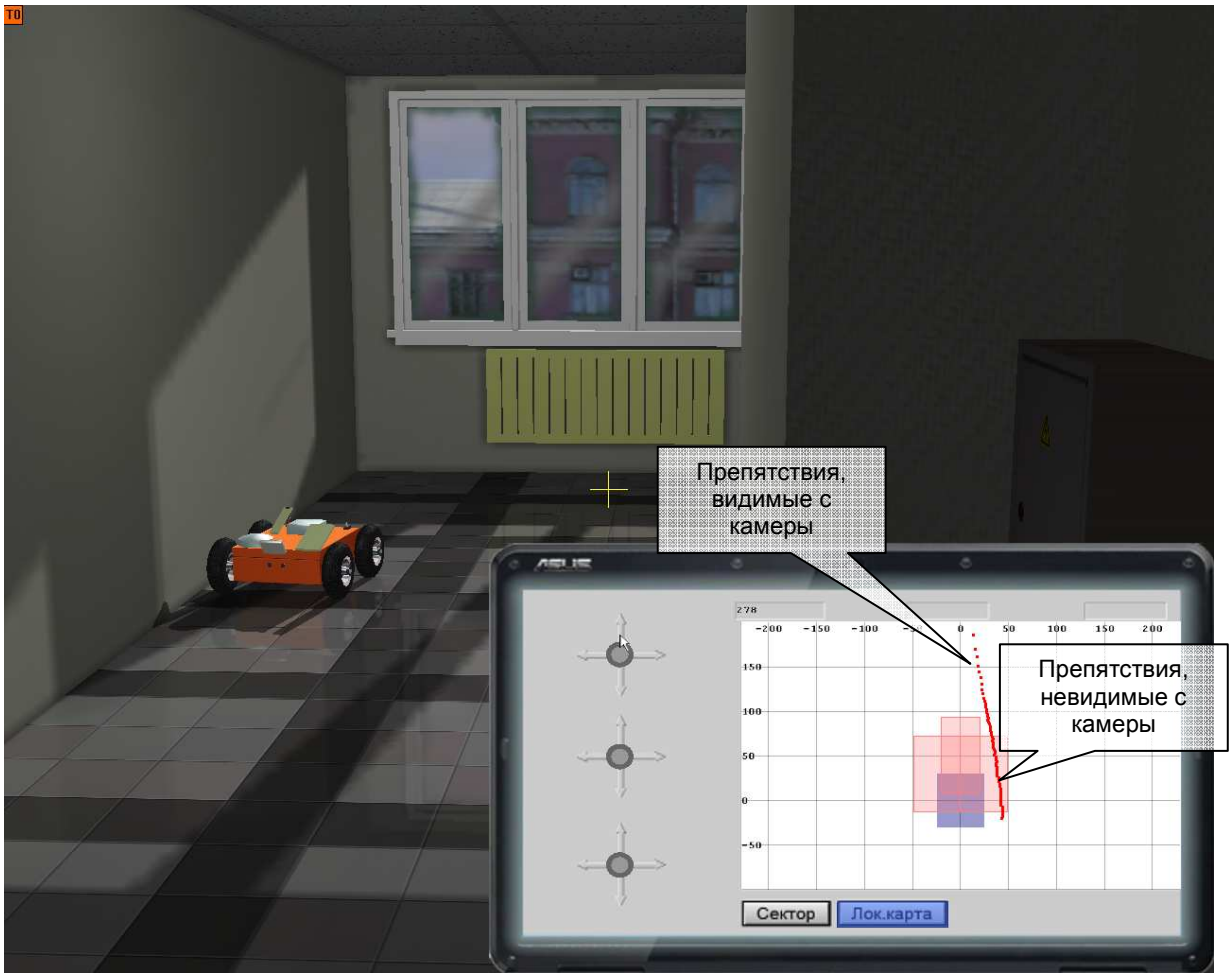

Рис. 31 Пример формирования роботом локальной карты местности в симуляторе Dyn-Soft RobSim 5

3.2.2. Зоны анализа на локальной карте местности

Цель формирования локальной карты местности заключается в оценке опасности различных направлений движения робота.

Самым простым способом определения опасности направления является определение наличия препятствий в той или иной области локальной карты местности. Для разметки таких областей применяется механизм установки *зон анализа*.

Зона анализа (обычно прямоугольной формы) – область локальной карты местности, в которой подсчитывается число элементов препятствий, попавших в нее. Считается, чем больше элементов препятствий попадает в зону, тем она опаснее. При наличии нескольких срезов локальной карты местности удобно назначать зоне анализа список слоев, анализ которых она будет производить.

В Dyn-Soft RobSim 5 зона анализа обозначаются на локальной карте местности бледно-красным цветом (Рис. 31).

Для мобильного робота зоны анализа рекомендуется задавать согласно схеме на Рис. 32. Слева от робота, захватывая часть пространства

робота, располагается левая зона анализа (зона 1). Справа от робота впритык к зоне 1 симметрично ей расположена правая зона анализа (зона 2). Прямо перед роботом, частично захватывая самого робота, находится передняя зона анализа (зона 3).

Таким образом, интеллектуальная система управления роботом обеспечивается информацией об опасности слева, опасности справа и опасности прямо.

В зависимости от решаемых задач, разработчик может назначить дополнительные зоны анализа и использовать информацию с них в интеллектуальной системе управления роботом.

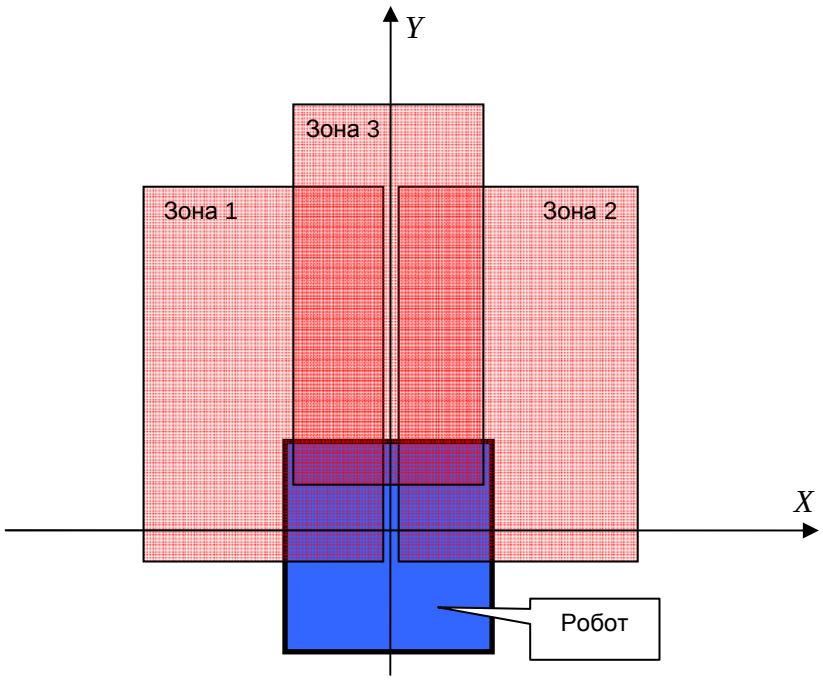

Рис. 32 Рекомендуемое размещение зон анализа на локальной карте местности

3.2.3. Виртуальные стены на локальной карте местности

Не стоит забывать о том, что препятствия в невидимых с системы очувствления областях локальной карты местности появляются только по мере движения робота. Но в начальный момент времени или после сброса локальной карты местности в невидимых областях локальной карты местности нет никаких препятствий, а зоны анализа (см. главу 3.2.2) пусты. Интеллектуальная система управления роботом в данном случае может ошибочно принять решение об отсутствии препятствий слева и справа от робота и предпринять попытку поворота, что в большинстве случаев приводит к столкновению с препятствиями в невидимой зоне.

Поэтому при сбросе локальной карты местности рекомендуется по обоим бортам робота сформировать из элементов препятствий виртуальные стены (Рис. 33).

Виртуальные стены не позволяют зонам анализа считать пространство слева и справа от робота пустым. По мере движения робота виртуальные стены передвигаются, как и все остальные элементы препятствий, постепенно замещаясь информацией о реальной обстановке вокруг робота.

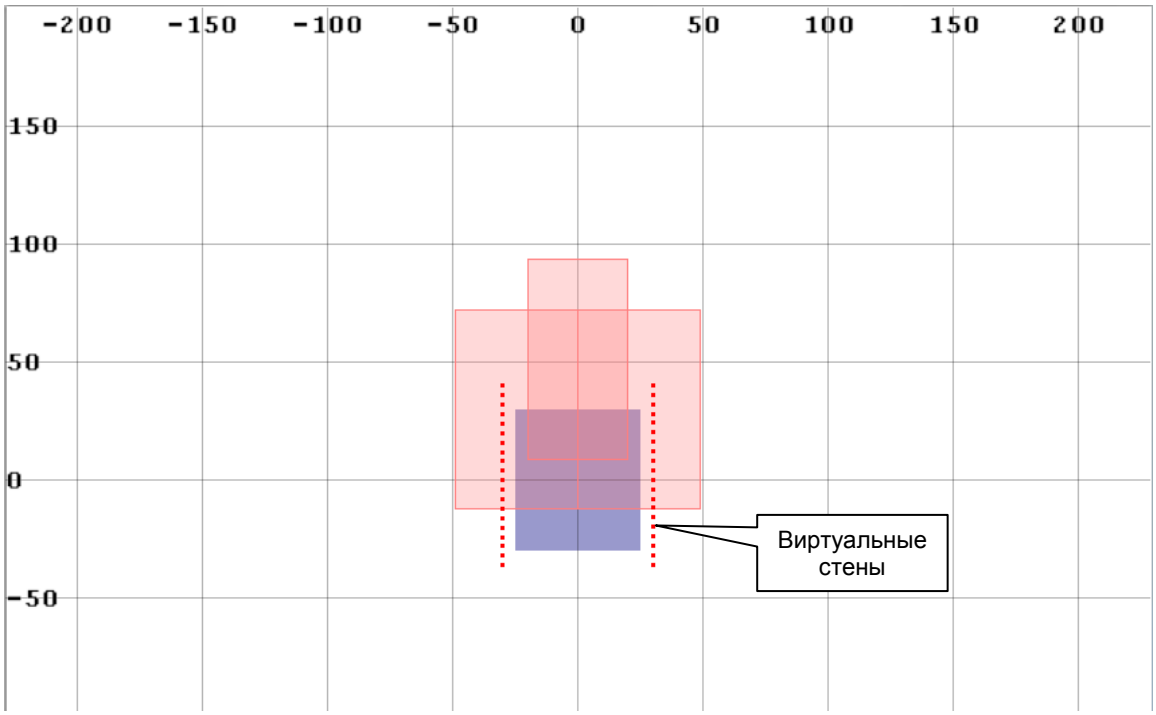

Рис. 33 Иллюстрация виртуальных стен на локальной карте местности

Также виртуальные стены на локальной карте местности рекомендуется формировать после затяжных (более чем на 20-30º) поворотов на месте при малой линейной скорости. Дело в том, что после таких поворотов теряется информация о наличии преград в направлении поворота (Рис. 34). Вовремя сформированные виртуальные стены в этом случае будут препятствовать столкновению с возможными препятствиями, находящимися в «мертвой зоне» поворота.

Механизм формирования виртуальных стен на локальной карте местности удобно совместить с функцией сброса локальной карты местности.

Сброс локальной карты местности совместно с формированием виртуальных стен необходим для очистки препятствий-призраков, обычно остающихся после внештатных перемещений или вращений робота (например, после столкновений), а также после езды по наклонным поверхностям (Рис. 35).

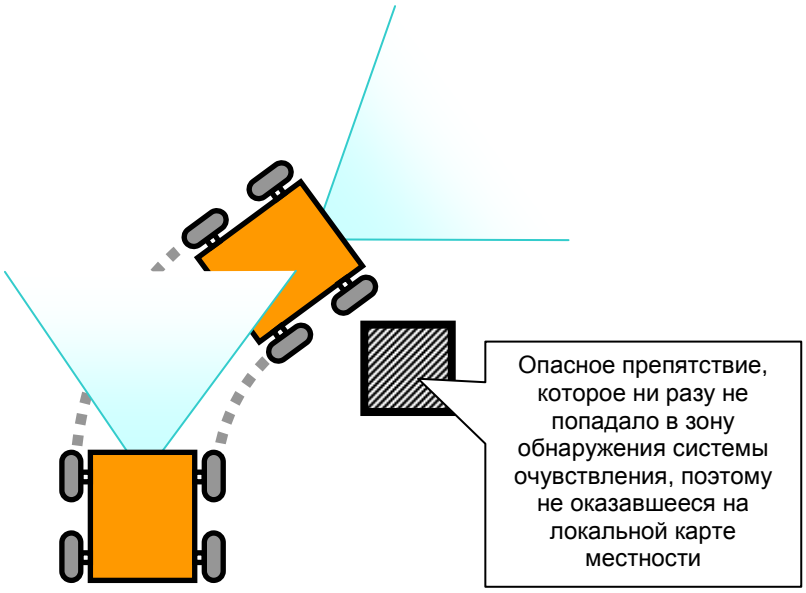

Рис. 34 Иллюстрация ситуации появления незамеченных опасных препятствий

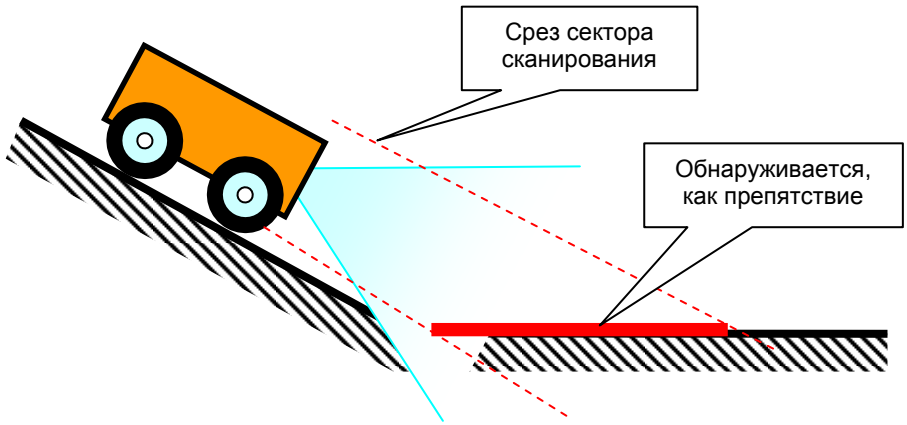

Рис. 35 Иллюстрация ошибочно определяемого препятствия при движении робота по наклонным поверхностям при формировании локальной карты местности

3.2.4. Формирование локальной карты местности в Dyn-Soft RobSim 5

Для формирования локальной карты местности в Dyn-Soft RobSim 5 имеется специальный блок «Построитель локальной карты местности», входящий в состав блоков редактора программного обеспечения бортовой ЭВМ (Рис. 36).

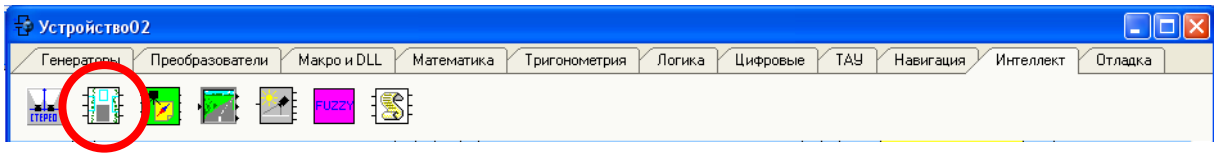

Рис. 36 Блок «Построитель локальной карты местности» в составе палитры блоков редактора программного обеспечения бортовой ЭВМ в Dyn-Soft RobSim 5

Используя данный блок, разработчик может создать структурную схему программного обеспечения бортовой ЭВМ, обеспечивающую формирование локальной карты местности на основе показаний систем стереозрения, дальномеров и прочих источников информации.

Данный блок позволяет настроить несколько каналов поступления входных данных. Для каждого такого канала создаются следующий набор входов:

- $sector_i$ или $dist_i$ – соответственно сектор сканирования (определяемый системой стереозрения) или показания дальномера (в сантиметрах) для *i*-ого канала.
- $write\_en_i$ – (тип данных bit) разрешение записи данных *i*-ого канала. Наличие «1» на этом входе разрешает запись данных из сектора сканирования или показания дальномера на локальную карту местности. Применяется, например, для запрета записи данных с несинхронизированной системы стереозрения во время движения робота.
- $sensorX_i$, $sensorY_i$ – двухмерные координаты положения (в сантиметрах) центра *i*-ого датчика системы очувствления в системе координат робота (ось *Y* направлена вперед, ось *X* – вправо) . Если входы не подключены, то положение датчика задается через внутреннюю настройку блока. Данные входы используются только при расположении датчика на подвижной или поворотной платформе.
- $sensorA_i$ – угол (в радианах) ориентации *i*-ого датчика системы очувствления в системе координат робота (по часовой стрелке, нулевой угол соответствует направлению вдоль оси *Y*). Если входы не подключены, то угол ориентации датчика задается через внутреннюю настройку блока. Данный вход используются только при расположении датчика на подвижной или поворотной платформе.

Кроме того, блок имеет входы, через которые поступает информация о перемещении элементов препятствий по карте местности:

- *v* – линейная скорость движения робота (м/сек);
- *w* – угловая скорость поворота робота (rad/сек);
- *reset* – (bit) сигнал сброса локальной карты местности.

На Рис. 37 приведен пример подключения блока «Построитель локальной карты местности» в структуру программного обеспечения бортовой ЭВМ.

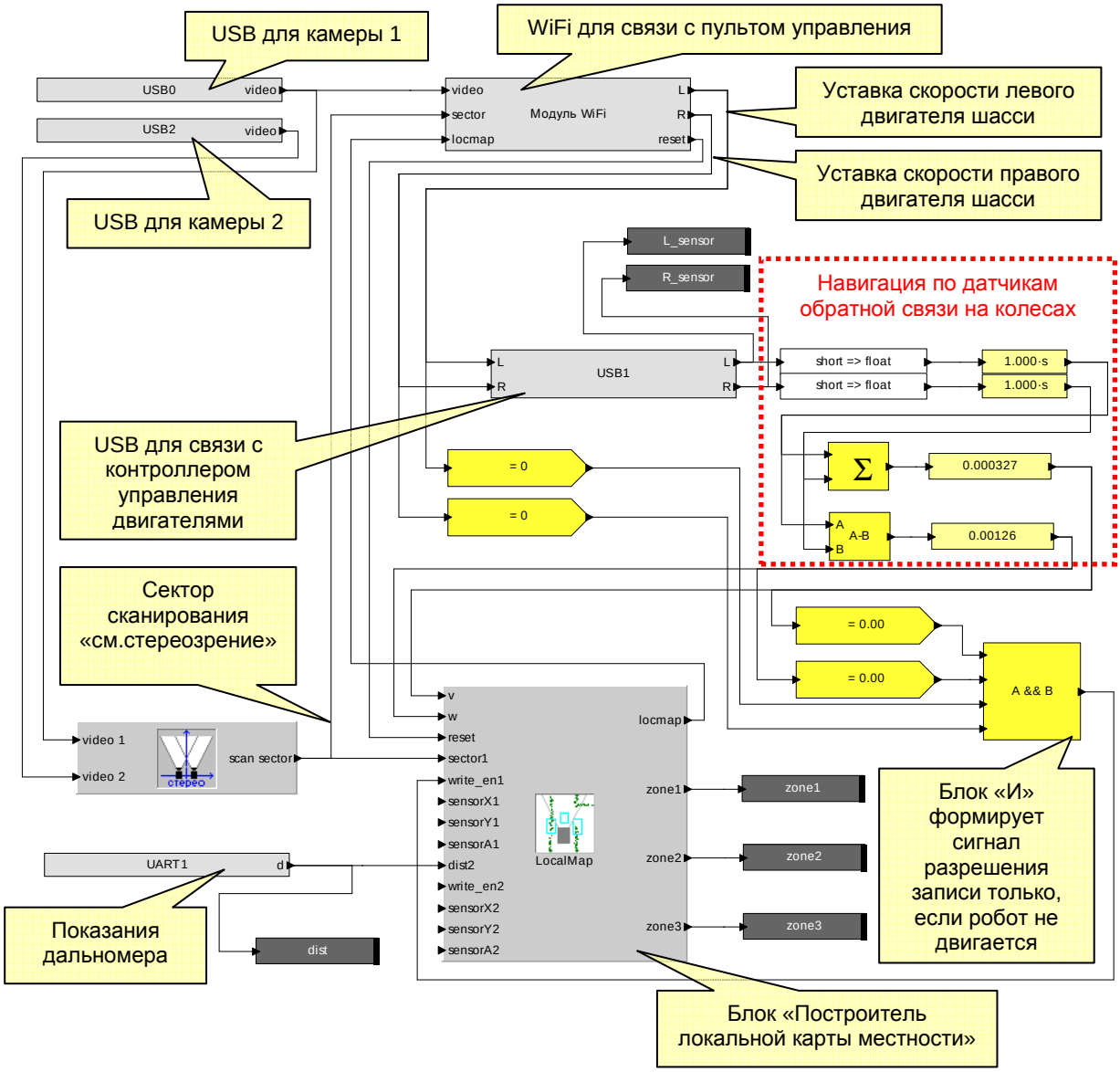

Рис. 37 Структура программного обеспечения бортовой ЭВМ для реализации алгоритмов локальной карты местности в Dyn-Soft RobSim 5

На рисунке ключевую роль играет блок «Построитель локальной карты местности», в настройках которого было установлено два источника информации (два сенсора). Один источник – система стереозрения (см. главу 3.1.3), формирующая сектор сканирования на основе видеоизображений с двух USB-камер (USB0 и USB2), другой – ультразвуковой дальномер, подключенный к порту UART1 бортовой ЭВМ (см. главу 2.2.7).

Как уже отмечалось в главе 3.1.2, во время движения робота несинхронизированная система стереозрения формирует неправильный сектор сканирования. Поэтому формирование локальной карты местности по системе стереозрения можно только тогда, когда робот стоит на месте. Поэтому в структуре программного обеспечения имеется механизм

формирования сигнала разрешения записи (блок «И»), который принимает значение 1 только, если на робота формируется нулевая уставка скорости для левого и правого борта, а также имеется нулевая линейная и угловая скорости. Сигнал разрешения записи поступает на вход write_en1 блока «Построитель локальной карты местности».

Обмен данными с контроллером управления двигателями робота производится через порт USB1. На него передаются уставки скорости левого и правого двигателя. В ответ контроллер управления по USB-порту передает на бортовую ЭВМ показание счетчика меток (типа short) левого и правого борта робота.

Используя алгоритмы навигации на основе показаний датчиков обратной связи на колесах (см. главу 4.3.3), формируются линейная скорость робота (м/сек), поступающая на вход *v* блока «Построитель локальной карты местности», и угловая скорость поворота робота (рад/сек), поступающая на вход *w* того же блока. Алгоритмы формирования локальной карты местности используют данные сигналы для перемещения и поворота элементов препятствий, в результате чего они передвигаются по карте.

Разработчику следует уделить особое внимание адекватности формирования данных скоростей. Чем точнее данные скорости соответствуют реальному перемещению робота, тем точнее формируется локальная карта местности. Если во время проведения предварительных испытаний разработчик замечает, что элементы карты, выходя за пределы зоны сканирования, быстро перемещаются по карте или наоборот скапливаются в одном месте, то линейная скорость робота формируется неправильно. Если во время поворота робота элементы препятствий существенно отстают или опережают реальный угол поворота робота, то угловая скорость робота формируется неправильно.

В приведенной структуре программного обеспечения бортовой ЭВМ на пульт оператора отправляются видеоизображение с левой камеры робота, сектор сканирования и локальная карта местности.

Для настройки блока «Построитель локальной карты местности» используется специальный встроенный редактор (Рис. 38). С помощью данного редактора разработчик может настроить параметры алгоритма формирования локальной карты местности.

На панели «размер карты» разработчик может задать размеры локальной карты местности. Карта формируется в координатах, относительно центра робота. Как уже отмечалось, не стоит делать карту слишком большой, т.к. из-за погрешности работы системы навигации на ней могут появляться препятствия-призраки. Чем меньше будет локальная карта местности, тем быстрее эти препятствия будут уходить за пределы карты и там исчезать.

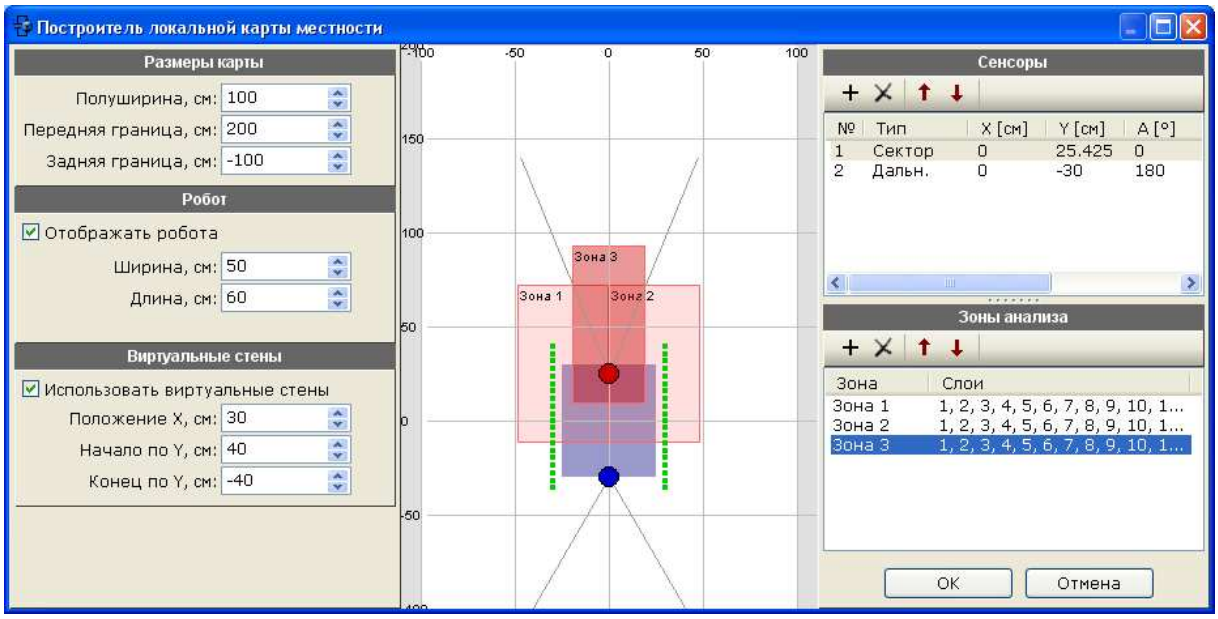

Рис. 38 Внешний вид диалогового окна редактора настройки блока «Построитель локальной карты местности»

На панели «робот» можно задать габаритные размеры робота. Данные габаритные размеры не влияют работу алгоритмов формирования локальной карты местности, а используются лишь для отображения локальной карты местности на экране. В заданных габаритных размерах на экране рисуется синий прямоугольник, изображающий робота. В принципе, с помощью соответствующей галочки можно отключить рисование этого прямоугольника при отображении карты.

На панели «Виртуальные стены» задаются параметры формирования виртуальных стен (см. главу 3.2.3) слева и справа от робота. Виртуальные стены будут формироваться всякий раз при сбросе локальной карты местности, если их не отключить соответствующей галочкой на данной панели.

На панели «Сенсоры» находится список сенсоров (каналов) блока «Построитель локальной карты местности». Разработчик может добавлять, удалять или изменять параметры сенсоров. При редактировании параметров сенсора открывается соответствующее диалоговое окно (Рис. 39, а). В данном окне пользователь может задать тип сенсора (сектор сканирования или дальномер), а также определить местоположение сенсора на роботе, его ориентацию относительно робота, угол обзора (сканирования), а также диапазон значений дальности.

Крайне важно правильно сформировать данные параметры, особенно угол обзора и диапазон дальности, т.к. данные величины формируют параметры видимой области локальной карты местности, которая при поступлении новых данных очищается.

Следует отметить, что положение X, Y формируется в сантиметрах относительно центра робота в его локальной системе координат (ось Y направлена вперед, а ось X – вправо). Угол поворота оптической оси сенсора задается в градусах по часовой стрелке. Нулевой угол соответствует направлению вперед.

Задавая положение и угол ориентации сенсора, разработчик переносит локальную систему координат сенсора, в которой он измеряет свои показания, в локальную систему координат робота.

Местоположение и ориентация сенсоров можно увидеть на схематическом отображении локальной карты местности в центре окна редактора.

а) б)

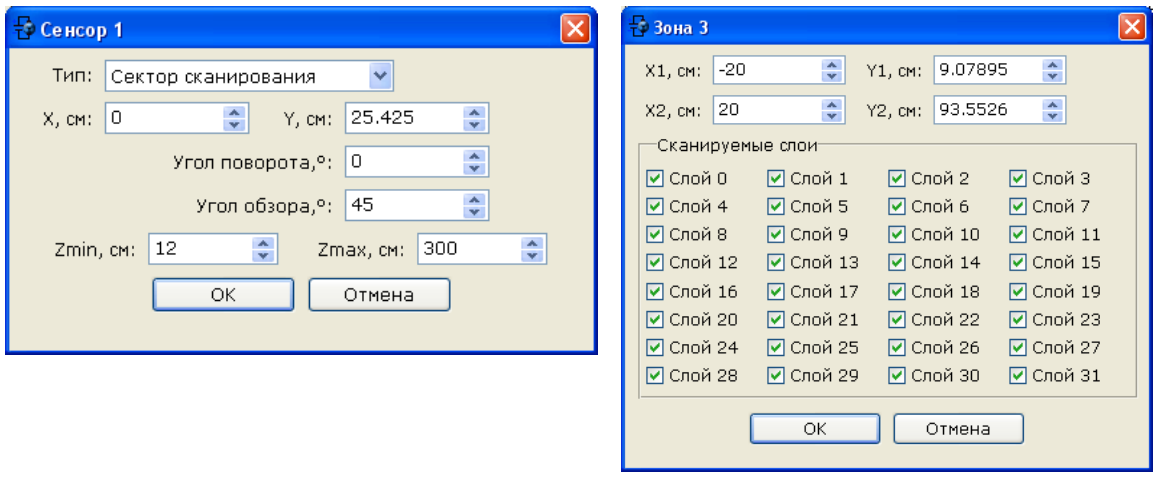

Рис. 39 Внешний вид вспомогательных диалоговых окон редактора настроек блока «Построитель локальной карты местности»: а) окно редактора сенсоров; б) окно редактора зон анализа.

На панели «Зоны анализа» имеется список всех зон анализа. Разработчик может добавлять, удалять или изменять параметры зон анализа. Следует отметить, что удобнее изменять координаты и местоположение зоны анализа в интерактивном режиме, путем перемещения мышкой соответствующей зоны по схематическому виду карты. Используя всплывающее меню на схематическом виде карты, можно создать копию или зеркальную копию зоны сканирования, что особенно актуально при создании левой и правой зоны анализа.

При редактировании свойств зоны анализа открывается диалоговое окно, изображенное на Рис. 39 (б). В данном диалоговом окне можно более точно задать координаты зоны, а также определить срезы (слои) (см. главу 3.1.3) локальной карты местности, которые будут учитываться при анализе опасности данной зоны.

Следует отметить, что число сенсоров и зон сканирования определяет число входов и выходов блока «Построитель локальной карты местности».

Для подтверждения всех изменений редактора следует нажать «OK», иначе никакие изменения свойств, в том числе добавление или удаление сенсоров и зон сканирования, не будут подтверждены.

На Рис. 40 приведена структура программного обеспечения пульта управления робота. Данная схема позволяет переключать источник видеоинформации, поступающий от бортовой ЭВМ робота по WiFi. Блок «Видео» в данной структуре может отображать или видеоизображение, или сектор сканирования, или локальную карту местности в зависимости от нажатой кнопки.

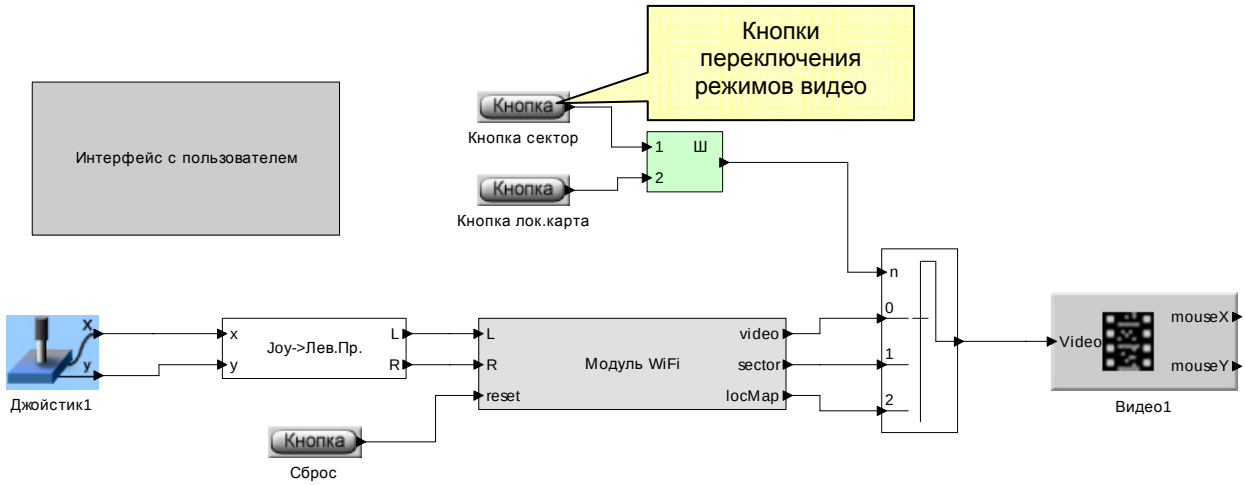

Рис. 40 Структура программного обеспечения пульта управления для реализации отображения обычного видео, сектора сканирования и локальной карты местности

Процесс проведения экспериментальных исследований с применением алгоритмов построения локальной карты местности показан на Рис. 41. На рисунке видно, что часть элементов препятствий была сформирована системой технического зрения. Однако та часть препятствий, которая находится справа от робота, явно выходит за пределы видимой области, однако за счет алгоритмов формирования локальной карты местности эти элементы препятствий были перенесены в невидимую область по мере движения робота.

Следует обратить внимание на относительную прямолинейность границ препятствий, отображаемых на локальной карте местности, и образующих естественное дополнение видимой области. Это говорит об адекватности работы системы навигации робота.

На рисунке в окне консоли отображаются опасности зон анализа (zone1, zone2, zone3) в зависимости от числа элементов препятствий, попадающих в эти зоны анализа. Важно, что эти опасности представляют собой скалярную величину, что дает возможность организовать автоматический процесс обхода препятствий для интеллектуального мобильного робота.

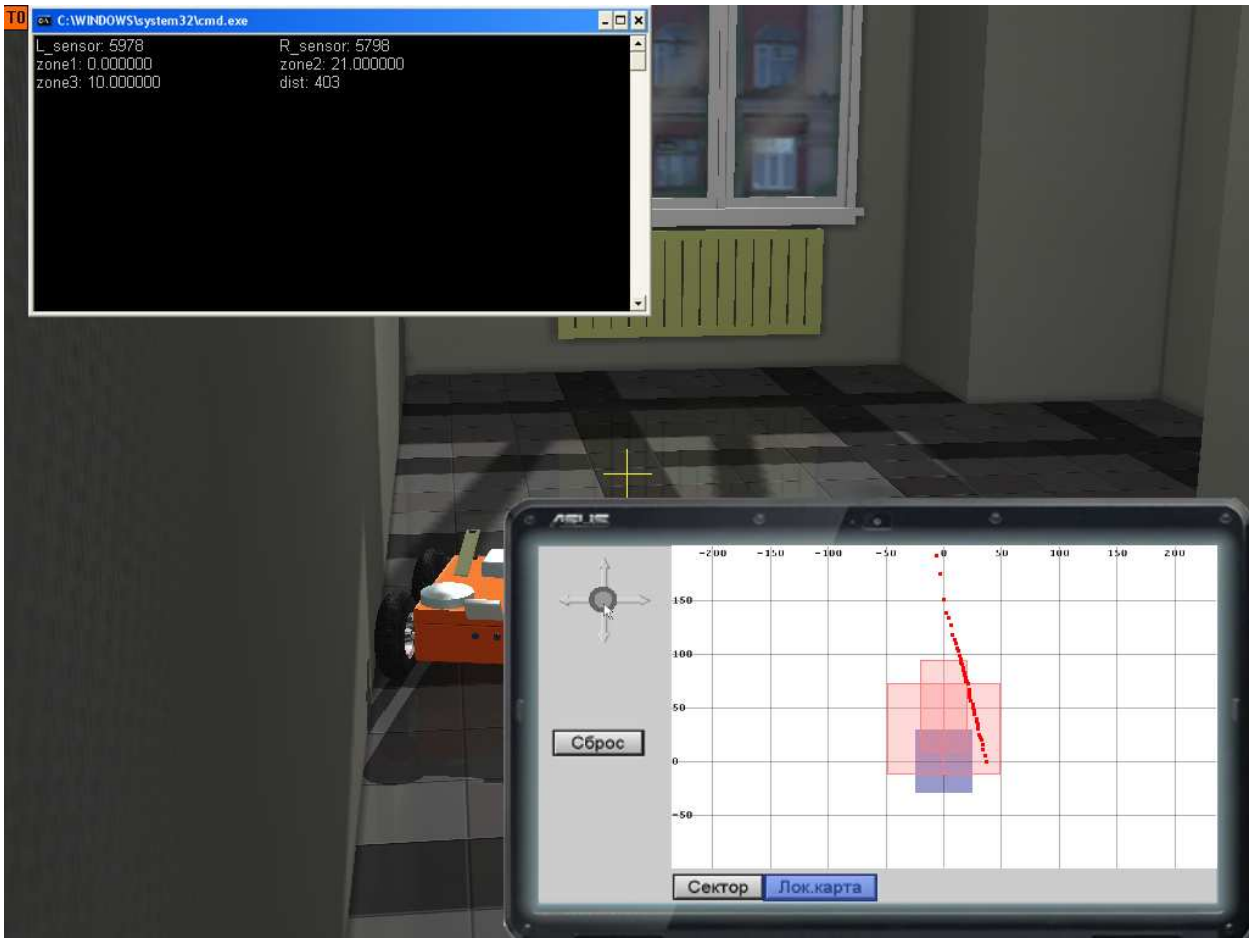

Рис. 41 Экранный снимок процесса проведения экспериментальных исследований работы алгоритмов построения локальной карты местности в Dyn-Soft RobSim 5

3.3.Детектор дороги

3.3.1. Принцип работы детектора дороги

Принцип работы детектора дороги заключается в анализе видеоизображения с одной камеры с целью выявления опасности одного из трех возможных направления движения робота.

Метод определения опасности направления основан на весьма сомнительном утверждении, что дорога содержит меньше контуров объектов, чем ее обочина. Поэтому опасность направления определяется по средней яркости оконтуренного изображения в одной из трех частях изображения (Рис. 42).

На рисунке в левой части изображены дороги, уходящие в разных направлениях. В правой части изображены оконтуренные изображения этих дорог. Цифрами обозначена средняя яркость оконтуренного изображения в процентах.

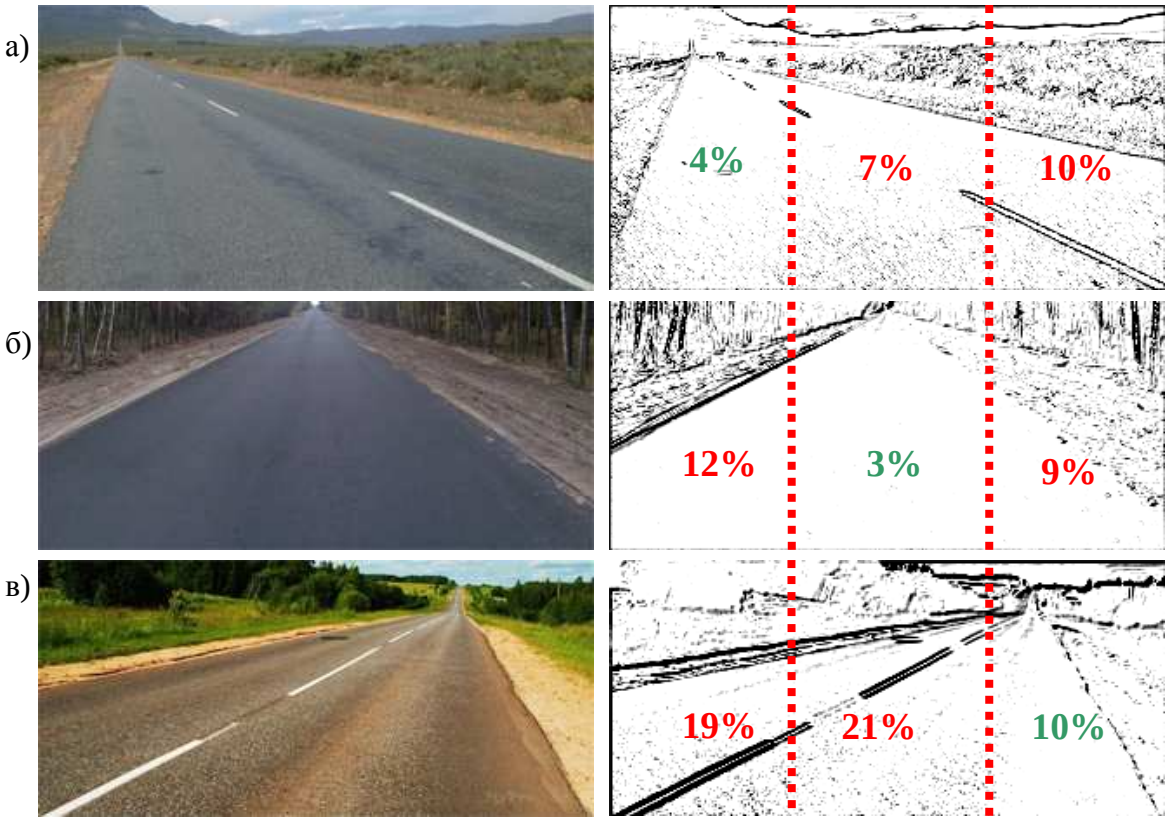

Рис. 42 Коэффициенты опасности дорог: а) дорога, уходящая налево; б) дорога, уходящая прямо; в) дорога, уходящая направо

Несложно заметить, что в направлении, в котором уходит дорога, контуров меньше, а, стало быть, средняя яркость оконтуренного изображения в этих частях изображения меньше.

Таким образом, выбирая направление с минимальной опасностью можно двигаться по дороге или в наиболее свободном направлении.

Следует отметить, что предложенный метод будет работать отнюдь не всегда, а лишь в тех случаях, когда на обочине дорог имеются кусты, трава, деревья и другие объекты, а сама дорога не покрыта трещинами в асфальте, ямами и прочими объектами.

В принципе, можно придумать массу различных контр примеров, при которых данный метод может дать сбой. Даже в Dyn-Soft RobSim 5 данный блок не всегда указывает правильное направление дороги в спорных ситуациях. Поэтому детектор дороги имеет смысл использовать совместно с другими системами.

3.3.2. Требования к сцене для использования детектора дороги в Dyn-Soft RobSim 5

Для адекватной работы блока «Детектор дороги» в Dyn-Soft RobSim 5 к виртуальной сцене предъявляются особые

требования, т.к. данный блок распознает не всякие дороги, нарисованные на сцене.

Блок «Детектор дороги» в Dyn-Soft RobSim 5 работает в режиме эмуляции. Т.е. он не обрабатывает реальное изображение с камеры, а оперирует с объектами 3D-сцены.

Поэтому для обеспечения работы данного блока, дорога на сцене должна обладать меньшим количеством полигонов, чем ее обочина. По краям дороги желательно установить различного рода объекты, типа столбиков, высоко полигональных ограждений, кустов, деревьев и т.п. Особенно чувствителен детектор дороги к наличию травы на сцене. Поэтому разработчику при создании сцены рекомендуется использовать карты размещения травы.

В составе дистрибутива Dyn-Soft RobSim 5, к сожалению, нет наиболее подходящего примера такой сцены, поэтому разработчику необходимо самостоятельно разрабатывать такую сцену.

3.3.3. Использование детектора дороги в Dyn-Soft RobSim 5

Мобильный робот, на который планируется установка блока «Детектор дороги», должен обладать камерой, направленной в направлении дороги. Желательно наличие небольшого наклона такой камеры, чтобы камера была направлена не на горизонт, а на пространство перед роботом.

Непосредственно сам блок «Детектор дороги» расположен на палитре блоков редактора программного обеспечения бортовой ЭВМ (Рис. 43).

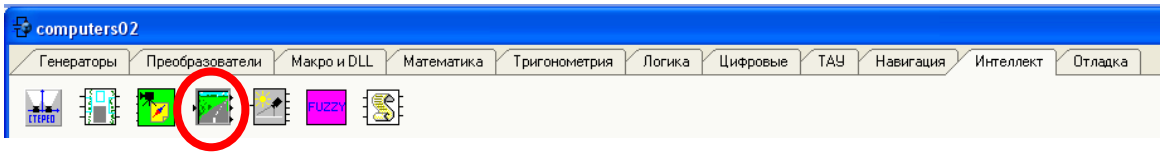

Рис. 43 Блок «Детектор дороги» в палитре блоков редактора программного обеспечения бортовой ЭВМ в Dyn-Soft RobSim 5

Для подключения данного блока необходимо на его вход подать сигнал с изображением с камеры, направленной на дорогу (Рис. 44), а на выходе блока будут формироваться три сигнала: опасность слева, опасность прямо и опасность справа. Каждый из выходных сигналов представляет собой тип данных float и изменяется от 0 до 1. Значение 1 означает опасное направление. Значение 0 означает свободное направление.

На Рис. 45 показан процесс проведения экспериментальных исследований работы блока «Детектор дороги», подключенного по приведенной схеме. Несложно заметить, что наличие травы слева и справа

от робота формирует достаточно высокие (порядка 0.9) уровни опасности этих направлений. В то время, как направление прямо, на котором нет ни травы, ни каких-либо других препятствий, формирует уровень опасности 0.

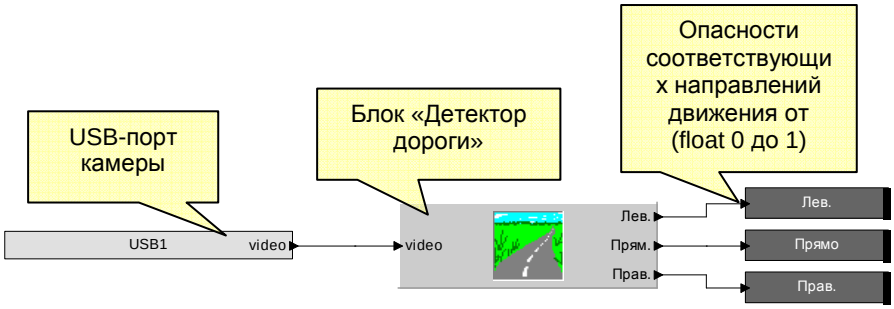

Рис. 44 Подключение блока «Детектор дороги» в редакторе программного обеспечения бортовой ЭВМ в Dyn-Soft RobSim 5

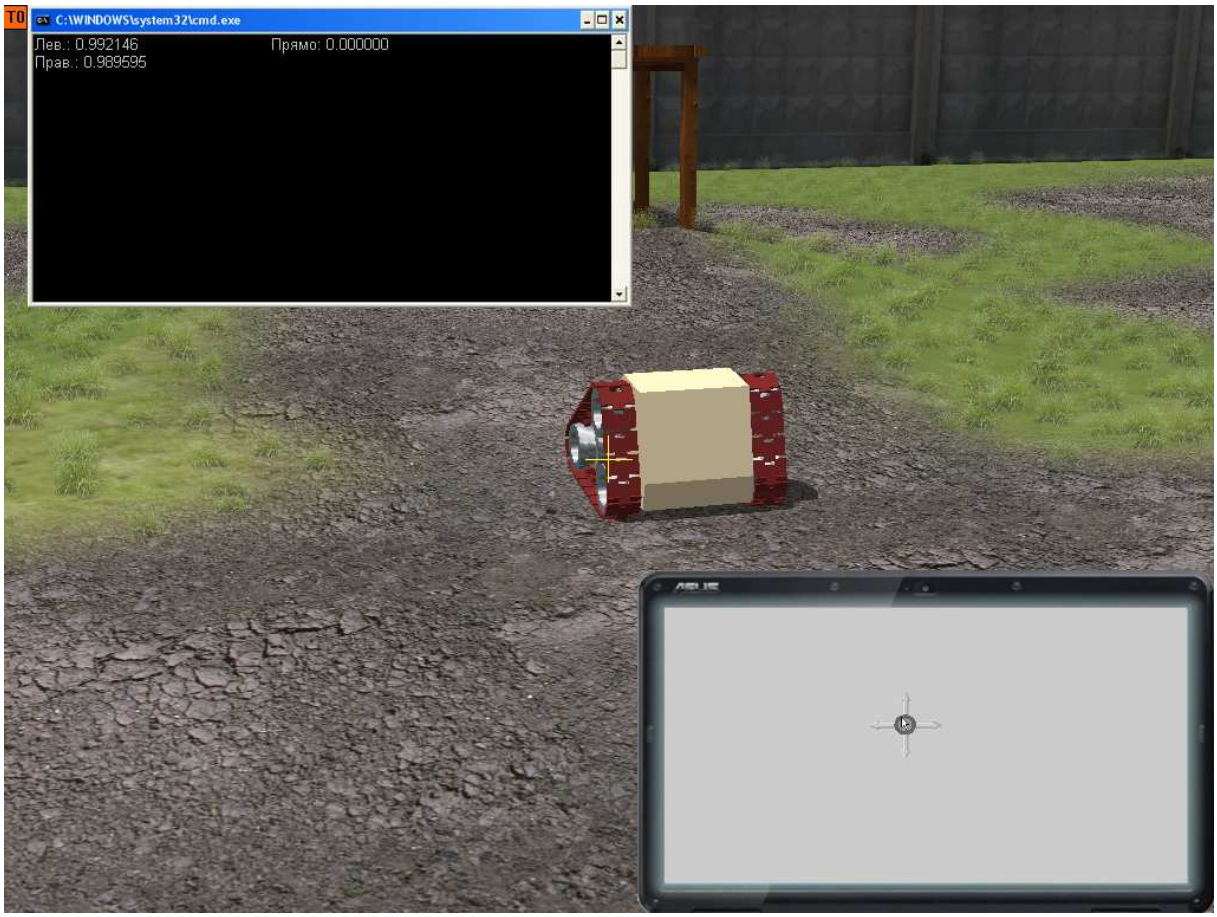

Рис. 45 Экранный снимок процесса проведения экспериментальных исследований работы блока «Детектор дороги»

Интеллектуальная система управления робота, основанная на «Детекторе дороги», должна анализировать уровни опасности и принимать решение о дальнейшем направлении движения. Удобно подобную систему сделать, например, по технологии нечеткой логики (см. главу 5.4).

3.4. Распознавание светоотражающей полосы

3.4.1. Светоотражающие полосы и их применение в робототехнике

Светоотражающие полосы для обеспечения навигации мобильных роботов применялись еще на заре развития робототехники. Данный способ навигации роботов является самым простым и надежным. Для его применения достаточно прочертить на полу полосу белой или иной краской.

Для распознавания светоотражающих полос ранние роботы использовали фотодиодные линейки и специальную подсветку. В настоящее время в связи с бурным развитием дешевых WEB-камер и вычислительных средств, для распознавания полосы удобнее использовать небольшую камеру с малым разрешением. Использование светодиодной линейки в настоящее время влечет излишние схемотехнические сложности и экономически нецелесообразно.

Применение цветной камеры для распознавания светоотражающей полосы дает возможность не только разметить маршрут движения робота, но и, используя различного рода цветовую маркировку, обозначить различную маршрутную информацию. Например, цветом полосы можно обозначить скорость движения на данном участке или места остановки. Также различными цветами можно обозначить пути к различным целевым точкам.

3.4.2. Подготовка сцены со светоотражающими полосами в Dyn-Soft RobSim 5

Симулятор Dyn-Soft RobSim 5 поддерживает механизмы распознавания светоотражающих полос в режиме эмуляции. Т.е. для распознавания полосы симулятор не обрабатывает реальное растровое изображение с камеры, а имитирует распознавание специальных объектов 3D Studio MAX, которым дополнительно помечена светоотражающая полоса.

Для обозначения на сцене светоотражающей полосы необходимо в 3D Studio MAX создать вдоль полосы один или несколько объектов типа Shape (двухмерные фигуры). Рекомендуется использовать ломанные линии (инструмент Line). Ломанную следует прочертить вдоль созданной в виде объекта или нарисованного в виде графического объекта 3D Studio MAX изображения полосы. В поле «название» созданной фигуры Shape следует вставить подстроку: «*STRIP*» или «*STRIP**идентификатор**», где

идентификатор – название или идентификатор данной полосы (аналог цвета при цветовой разметки полосы).

Например:

```
*STRIP*line01
```

или

```
*STRIP*WHITE*01
```

Данная вставка в название объекта может быть, как в начале, так и в любом другом месте строки. Линии, обозначающие полосы, могут быть как самостоятельным объектами 3D Studio MAX, так и быть частями (Spline) одного объекта типа Shape.

3.4.3. Распознавание светоотражающих полос в Dyn-Soft RobSim 5

Для распознавания светоотражающих полос мобильный робот должен обладать камерой, направленной вниз на светоотражающую полосу. Камеру не рекомендуется устанавливать совсем низко, т.к. у нее будет весьма ограниченное поле зрения.

Для распознавания полосы на изображении с этой камеры в структурную схему программного обеспечения бортовой ЭВМ следует установить один или несколько (по числу идентификаторов светоотражающих полос) блоков «Блок распознавания светоотражающей полосы». Расположение этого блока на палитре блоков редактора программного обеспечения показано на Рис. 46.

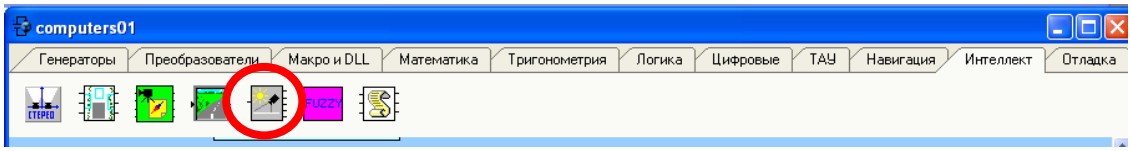

Рис. 46 «Блок распознавания светоотражающей полосы» в палитре блоков редактора программного обеспечения бортовой ЭВМ в Dyn-Soft RobSim 5

Подключение блока распознавания светоотражающей полосы показано на Рис. 47. На рисунке сигнал с USB-камеры, подключенной к порту USB2, подается на вход этого блока.

В настройках свойств блока распознавания светоотражающей полосы задается идентификатор светоотражающих полос, на распознавание которых настроен данный блок. Например, если светоотражающие полосы имеют название «*STRIP*WHITE**xx*», то в свойствах блок следует указать идентификатор «WHITE». Если идентификатор в свойствах блока не указан, то блок распознает все типы полос.

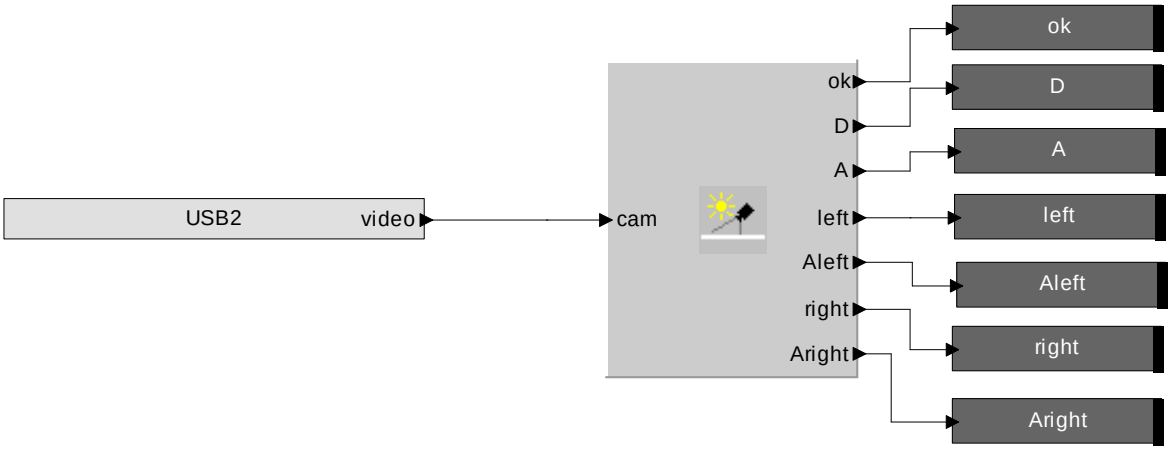

Рис. 47 Пример подключения блока распознавания светоотражающей полосы в структуру программного обеспечения бортовой ЭВМ в Dyn-Soft RobSim 5

На выходе блока распознавания светоотражающей полосы формируются следующие сигналы:

- ok – (бит) признак обнаружения полосы. Если данный сигнал равен 0, то состояние всех остальных выходов блока недействительно.
- D – (float) смещение светоотражающей полосы относительно центра кадра. Изменяется от -1 до +1. Значение -1 соответствует смещению к левой границе кадра. Значение +1 соответствует смещению полосы к правой границе кадра. Если полоса ровно по центру кадра, то значение сигнала D равно 0. Смещение считается по точке пересечения полосы с горизонтальной прямой, проходящей через центр кадра. Если сигнал ok равен 0, то сигнал D тоже равен 0.
- A – (float) угол поворота полосы в радианах при ее проекции на кадр видеоизображения. Нулевое значение соответствует вертикальному положению светоотражающей полосы на кадре. Положительное направление считается по часовой стрелке. Следует обратить внимание, что угол считается именно относительно проекции полосы на кадр видеоизображения, а не относительно самого робота. Если сигнал ok равен 0, то сигнал A тоже равен 0.
- left – (bit) признак наличия левого ответвления от полосы.
- Aleft – (float) угол направления левого ответвления от полосы (в радианах). Положительное направление – по часовой стрелке, значение 0 соответствует направлению вверх на карде изображения. В случае отсутствия левого ответвления (если сигнал left равен 0), значение сигнала Aleft совпадает со значением сигнала A.
- right – (bit) признак наличия правого ответвления от полосы.

- Aright – (float) угол направления правого ответвления от полосы (в радианах). Положительное направление – по часовой стрелке, значение 0 соответствует направлению вверх на карде изображения. В случае отсутствия левого ответвления (если сигнал right равен 0), значение сигнала Aright совпадает со значением сигнала A.

При использовании блока разработчику следует обратить внимание на способы формирования блоком правых и левых ответвлений. Направлению прямо соответствует такое ответвление полосы, у которого меньший по модулю угол отклонения по вертикали. Таким образом, перекресток прямо и направо под 90º при угле поворота робота более 45º будет распознаваться как перекресток прямо и налево (Рис. 47).

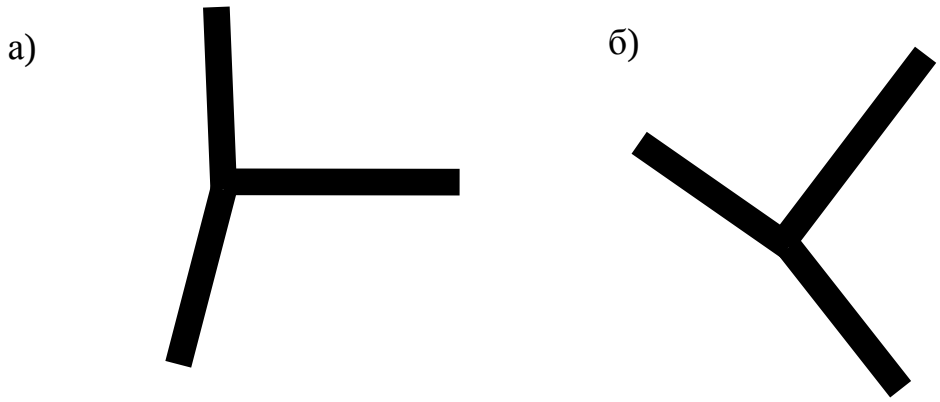

Рис. 48 Иллюстрация способов распознавания перекрестков: а) перекресток прямо и направо; б) тот же перекресток под углом поворота камеры более 45º, который распознается как налево и прямо

Поэтому система управления робота должна учитывать возможность появления подобного рода неоднозначностей.

Кроме того, при использовании блока распознавания светоотражающей полосы следует учитывать, что информацию о поворотах блок формирует лишь те моменты времени, когда перекресток находится в поле зрения камеры. Поэтому разработчику следует предпринять меры по продлению импульсов наличия поворотов или использовать для этого триггеры.

3.4.4. Проведения экспериментальных исследования системы распознавания светоотражающих полос в Dyn-Soft RobSim 5

Процесс проведения экспериментальных исследований робота, оснащенного системой распознавания светоотражающих полос, показан на Рис. 49. Следует обратить внимание, что робот на рисунке повернут по отношению к полосе. В окне консоли на экранном снимке отображаются показания сигналов «Блока распознавания светоотражающей полосы».

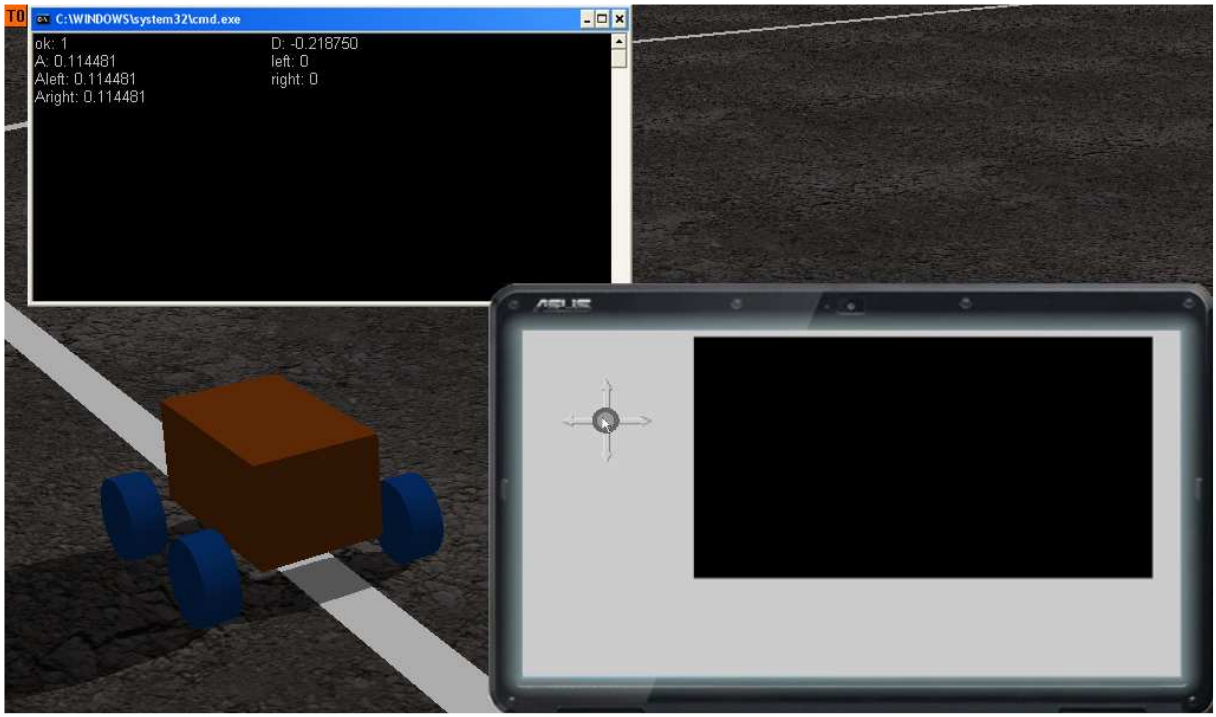

Рис. 49 Иллюстрация процесса проведения экспериментальных исследований системы распознавания светоотражающих полос в Dyn-Soft RobSim 5

Для тестовых экспериментальных исследований использовалась сцена «Scene1», входящая в состав дистрибутива программного комплекса Dyn-Soft RobSim 5. Данная сцена является единственной сценой стандартной сценой, содержащей светоотражающую полосу. Предполагается, что виртуальную сцену под конкретные цели и задачи робота пользователь должен разрабатывать самостоятельно.

4. Системы навигации интеллектуальных мобильных роботов

4.1. Системы радионавигации. GPS, ГЛОНАСС, псевдоспутники

4.1.1. Системы глобальной навигации (GPS/ГЛОНАСС)

В настоящее время в мире применяется две системы глобальной навигации: GPS и ГЛОНАСС. К 2020 году планировалось также завершение развертывания европейской системы глобальной навигации GALILEO.

Системы GPS и ГЛОНАСС используют принцип *радионавигации*, который заключается в том, что на орбите Земли движется несколько космических аппаратов, которые излучают специальные *радионавигационные сигналы*. У потребителя имеется *навигационный приемник*, который, получив как минимум 4 радионавигационных сигналов с разных спутников, определяет, так называемые, *псевдодальности* до спутников и их *псевдоскорости (допплеровские сдвиги частоты)*.

Важно, что все 4 спутника должны находиться в прямой видимости для приемной антенны GPS/ГЛОНАСС, и переотражения сигнала не допускаются. Поэтому работа GPS/ГЛОНАСС внутри помещений невозможна.

Зная 4 псевдодальности и декартовы координаты спутников, можно решить, так называемую, *навигационную задачу*, т.е. определить собственные декартовы координаты X, Y, Z, а также Δt – погрешность собственных часов. Кроме того, зная 4 псевдоскорости и векторы скорости 4 спутников, можно определить вектор собственной скорости V_x, V_y, V_z и погрешность собственного генератора частоты.

Полная космическая группировка GPS состоит из 32 космических аппаратов, космическая группировка ГЛОНАСС состоит из 24 спутников. Ряд спутников той и другой системы имеются на орбите в качестве резервных.

Следует отметить большой недостаток системы GPS. Она практически не покрывает территорию за северным и южным полярным кругом, поэтому навигация по GPS в северных морях в принципе невозможна. По наблюдениям водителей, использующих навигаторы, проблемы с приемом сигналов GPS уже начинаются с широты, чуть севернее Санкт-Петербурга.

Орбиты спутников ГЛОНАСС выше, и рассчитаны таким образом, что покрывают всю территорию Земли. Однако самих спутников ГЛОНАСС меньше, поэтому в большинстве случаев потенциально возможен прием сигналы только от 4 спутников.

4.1.2. Принцип работы системы радионавигации

Система глобальной навигации используется следующий механизм.

У каждого спутника имеются высокоточные часы (высокостабильный генератор частоты). За синхронностью этих часов следят специальные наземные станции. Точность синхронизации составляет пикосекунды. Все спутники синхронно друг с другом излучают кодовую посылку, так называемую *псевдослучайную дальномерную последовательность* (ПСПД) с наложенной поверх нее *навигационной информацией*. При излучении используется фазовая модуляция.

ПСПД для ГЛОНАСС содержит 511 символов и период повторения 1 мс. Навигационная информация содержится в том же сигнале, но промодулирована на гораздо более низкой частоте (100 Гц – навигационная информация, 200 Гц – метка времени). Навигационная информация передается кадрами по 30 секунд. Каждый такой кадр содержит ежесекундные *метки времени*, *оперативную информацию*, повторяемую каждый кадр, и часть *неоперативной информации*. Каждые 5 кадров навигационной информации представляют собой, так называемый, *суперкадр*. За один суперкадр передаются все части неоперативной информации. Т.о. суперкадр повторяется через 2.5 минуты.

Оперативная информация содержит информацию о том спутнике, который передает данную информацию, в частности, координаты спутника на орбите, его скорость, ускорение и прочую служебную информацию. Неоперативная информация содержит информацию о других спутниках орбитальной группировки (так называемый, *альманах*).

Альманах необходим приемнику для быстрого перехода с одного спутника на другой, который возможно еще не появился в зоне прямой видимости приемника. Однако в момент появления, его координаты, скорость и ускорения будут уже известны навигационному приемнику, что позволит использовать данный спутник для навигации без ожидания приема от него оперативной информации.

Система ГЛОНАСС используют частотное разделение, т.е. каждый спутник излучает сигнал на своей частоте, однако из-за ограничения числа частотных каналов спутники, находящиеся по разные стороны Земли могут работать на одной частоте. ПСПД всех спутников одинаковая.

Спутники системы GPS излучают сигналы на одной частоте, однако используют фазово-кодовую модуляцию сигналом ПСПД. У каждого спутника GPS своя ПСПД. Для дешифрования информации потребитель должен знать ПСПД спутника, чтобы «вытащить» полезный сигнал из-под

«уровня шума» на данной частоте. Такой подход автоматически делает невозможным использование военного канала GPS, сигналы которого промодулированы секретной ПСПД.

Аналогично с системой ГЛОНАСС спутники системы GPS также передают навигационную информацию, которая содержит оперативную информацию, неоперативную информацию и метки времени. Кадр навигационной информации также передается за 30 секунд, а за 2.5 минуты передается один суперкадр. В отличие от ГЛОНАСС оперативная и неоперативная информация GPS содержит не положение, скорость и ускорение спутника на орбите, а фазу движения спутника по орбите и ее поправки. Предполагается, что в навигационный приемник GPS заложены орбиты спутников GPS, а его вычислитель имеет возможность вычислять координаты и вектор скорости спутника на орбите, имея информацию о фазе спутника на орбите и ее поправки.

Новые спутники системы ГЛОНАСС-М также использовать фазово-кодовую модуляцию.

Навигационный приемник принимает сигналы спутников и с помощью *коррелятора* определяет фазу ПСПД относительно собственных часов, момент появления в навигационной информации метки времени (относительно собственных часов), а также допплеровский сдвиг частоты. Фаза ПСПД (точное приближение без учета периода повторения ПСПД) и момент появления метки времени (грубое приближение) позволяют определить псевдодальность до *i*-ого спутника (T_i). Псевдодальность измеряется в секундах и определяется относительно часов навигационного приемника. Причем на этапе измерения величина рассогласования Δt собственных часов приемника относительно спутникового времени неизвестна. Поэтому каждая псевдодальность содержит неизвестное время Δt:

$$T_i = t_i + \Delta t$$

Расстояние до *i*-ого спутника определяется, исходя из скорости света (c):

$$D_i = \left(T_i - \Delta t\right) \cdot c$$

Зная, как минимум, 4 псевдодальности, а также координаты соответствующих спутников S_i (S_{ix}, S_{iy}, S_{iz}), можно вычислить собственные координаты P (P_x, P_y, P_z) приемника и величину Δt, решив систему уравнений:

$$\begin{cases} \sqrt{(S_{1x}-P_x)^2+(S_{1y}-P_y)^2+(S_{1z}-P_z)^2} = (T_1-\Delta t)\cdot c \\ \sqrt{(S_{2x}-P_x)^2+(S_{2y}-P_y)^2+(S_{2z}-P_z)^2} = (T_2-\Delta t)\cdot c \\ \sqrt{(S_{3x}-P_x)^2+(S_{3y}-P_y)^2+(S_{3z}-P_z)^2} = (T_3-\Delta t)\cdot c \\ \sqrt{(S_{4x}-P_x)^2+(S_{4y}-P_y)^2+(S_{4z}-P_z)^2} = (T_4-\Delta t)\cdot c \end{cases}$$

Навигационная задача заключается в решении данной системы уравнений. Следует отметить, что алгебраического решения данного уравнения нет, поэтому приходится использовать итерационные алгоритмы. Хорошо зарекомендовал себя метод наименьших квадратов, позволяющий в 2-3 итерации достаточно близко приблизиться к точному решению.

Аналогичным образом составляются уравнения по скорости. Зная допплеровский сдвиг частоты 4 спутников, расстояния до них, а также вектора скорости спутников можно вычислить вектор собственной скорости приемника и рассогласование частоты собственного генератора частоты.

Различают «холодный старт» и «горящий старт» навигационного приемника.

«Холодный старт» подразумевает первый запуск навигационного приемника после его включения. При этом пока приемник не принял хотя бы одного кадра навигационной информации, он не сможет начать работу, т.к. не знает координаты спутников на орбите. Как уже отмечалось, кадр навигационной информации передается за 30 секунд. Поэтому, в лучшем случае, после холодного запуска приемник может начать работу только спустя 30 секунд, а в худшем через 1 минуту.

«Горящий старт» предполагает, что навигационный приемник находился в режиме пониженного потребления питания, при этом не решает навигационной задачи, но принимает сигналы навигационных спутников. Поэтому после «горячего старта» навигационная информация каждого спутника уже имеется, и приемник может сразу же начать работать. Обычно «горячий старт» навигационного приемника требует лишь время для запуска навигационного вычислителя.

Некоторые производители навигационного оборудования производят поэтапный «горячий старт». Предполагая, что часы навигационного прибора относительно точные и зная орбиты спутников, вычисляют положение спутников на орбите в данный момент времени и производят «горячий старт». После приема кадра навигационной информации часы навигационного прибора обновляются, что дает возможность более точно вычислить альманах (положение всех спутников на орбите), а, следовательно, более точно определить координаты приемника.

4.1.3. Источники погрешности систем радионавигации

Системы радионавигации имеют 5 основных источника погрешности:

- ионосферная погрешность;
- тропосферная погрешность;
- эфемеридная погрешность;
- погрешность часов приемника;
- геометрический фактор.

Ионосферная и тропосферная погрешности связана с тем, скорость распространения радиоволн в ионосфере и тропосфере Земли несколько отличается от скорости света в вакууме. В силу того, что состояние ионосферы и тропосферы Земли нестабильно, поэтому радиодальность до спутника содержит случайную погрешность, вызванную влиянием ионосферы Земли на время распространения радиосигнала.

Эфемеридная погрешность связана с погрешность определения координат самих спутников на их орбитах. По признанию космических баллистиков, координаты спутника на орбите можно определить с точностью лишь до 5-10 метров!

Погрешность часов приемника связана с нестабильность генератора приемника. В силу использования на навигационном приемнике неточного тактового генератора, возникает погрешность определения псевдодальности разных спутников.

Геометрический фактор (PDOP) определяет погрешность определения координат, возникающую в связи с наличием вышеперечисленных погрешностей определения расстояния до спутника. Некоторое взаимное расположение спутников приводит к вырождению системы уравнений навигационной задачи или делает решение нестабильным. Например, если все 4 спутника будут располагаться на одной прямой вместе с потребителем, то определить координаты потребителя будет вообще невозможно (множество решения). И чем ближе спутники приближаются к такому расположению, тем более нестабильным становится решение навигационной задачи (т.е. малая погрешность определения одной псевдодальности дает большую погрешность определения местоположения потребителя). По сути, *геометрический фактор – это отношение абсолютной погрешности вычисления координат приемника к погрешности измерения расстояния до спутника*.

Различают полный геометрический фактор, геометрический фактор в плоскости и по высоте. Поэтому навигационный приемник обычно выдает на выходе значение геометрического фактора. Если геометрический фактор больше 3-4, то решению навигационной задачи не доверяют.

В результате суммарная погрешность определения координат при помощи GPS/ГЛОНАСС составляет 10-30 метров.

Владельцы дорожных навигаторов могут с этой погрешностью поспорить, утверждая, что их навигатор всегда точно определяет местоположение их автомобиля, и они такой погрешности не замечали. Однако не стоит забывать, что дорожные навигаторы всегда привязывают местоположение автомобиля к дороге, поэтому у водителя складывается впечатление о точности работы прибора.

Существует множество способов борьбы с погрешностью систем глобальной навигации.

Во-первых, существуют методы уменьшения геометрического фактора за счет приема сигнала от большего числа спутников (более чем 4). Обычно приемники, принимающие сигналы одновременно от спутников GPS и спутников ГЛОНАСС, имеют более высокую точность.

Во-вторых, использование GPS/ГЛОНАСС в дифференциальном режиме. При этом используют два навигационных приемника. Один из них стационарный, его помещают в заранее известных координатах вблизи зоны функционирования другого мобильного приемника. В силу того, что координаты стационарного приемника известны, их можно вычесть из координат, определяемых приемником, получив тем самым погрешности измерения. Эти погрешности отправляются на мобильный приемник, устраняя тем самым ионосферную/тропосферную и эфемеридную погрешности. Точность измерения при этом существенно возрастает и составляет порядка 10 см.

Для работы в дифференциальном режиме существуют специальные наземные станции коррекции, которые передают на все наземные приемники дифференциальные поправки. К ним относятся:

- WAAS (США);
- EGNOS (Европа);
- СКНОУ (Украина);
- и другие.

Каждая такая станция имеет свой протокол и способ передачи этих дифференциальных поправок потребителям. При этом приемник должен обладать возможностью приема этих дифференциальных поправок по данному протоколу.

К сожалению, станций дифференциальных поправок на территории России почти нет.

В-третьих, существует способ определения координат по сигналам спутниковой навигации, связанный с накоплением статистики за большой интервал времени (например, за сутки). Однако подходит он только для стационарных объектов. При статистической обработке показаний измерения прибора одних и тех же координат можно добиться погрешности определения местоположения менее 1 мм. Однако данный способ не подходит для мобильных объектов.

4.1.4. Использование систем глобальной навигации в военное время

Система GPS целиком управляется США. При этом до 2000 года в сигналы гражданского GPS умышленно вводилась погрешность, не позволяющая определять координаты потребителя точнее 100 м. После 2000 года режим ограничения был снят, но США не брезгуют повторным вводом этого режима над локальными территориями во время своих вооруженных конфликтов. Известно, что во время войны в Ираке, а также во время грузинского конфликта в 2008 году США вводили эти погрешности в работу GPS.

Приказ министерства обороны России запрещает использовать в военной технике систем глобальной навигации на основе GPS.

Известно, что современные вооруженные конфликты уже давно не проходят без применения средств радиоэлектронной борьбы (РЭБ). Средствами РЭБ оснащаются все военные самолеты, используются специальные постановщики помех наземного и воздушного базирования, все они предназначены для подавления радиосвязи и командных линий управления военной техникой противника, в т.ч. высокоточного оружия, головок самонаведения ракет и т.п. Причем в первую очередь подавляются сигналы системы глобальной навигации.

Сигналы спутниковой навигации достаточно слабые и для их подавления от постановщика помех не требуется высокой мощности. В сети интернет продается даже устройства, которые от автомобильного прикуривателя подавляют GPS/ГЛОНАСС в радиусе 100 м.

Поэтому в военной технике использование систем глобальной навигации неприемлемо.

4.1.5. Локальные радионавигационные системы и наземное дополнение GPS/ГЛОНАСС (псевдоспутники)

Источникам радионавигационного сигнала не обязательно находиться на орбите Земли. Можно расположить несколько формирователей радионавигационного сигнала (псевдоспутников) на Земле и в некотором небольшом районе создать локальное радионавигационное поле.

В отличие от космических аппаратов, псевдоспутники не обладают ни эфемеридной погрешностью (т.к. расположены неподвижно в известных координатах), ни ионосферной погрешностью (т.к. сигнал от них не проходит через верхние слои атмосферы). Уровень сигнала псевдопутников может быть значительно выше, чем уровень сигнала GPS/ГЛОНАСС, что затрудняет его подавление естественными или преднамеренными помехами.

Для организации навигации по локальному радионавигационному полю должно быть не менее трех псевдоспутников (для навигации в плоскости) и четырех (для навигации в трехмерном пространстве). Правда, для надежной навигации в трехмерном пространстве псевдоспутники должны располагаться на разной высоте и не лежать в одной плоскости. Причем для обеспечения приемлемого геометрического фактора разброс высот должен быть существенный, а если быть точным, то расстояние от каждого псевдоспутника до плоскости, проведенной через 3 других спутника должно быть примерно на порядок выше, чем требуемая точность по высоте.

Применяется также режим наземного дополнения, который подразумевает работу по спутникам глобальной системы навигации и одному или нескольким наземным псевдоспутникам. При этом точность определения координат приемника значительно взрастает, особенно по высоте.

Однако для использования пседоспутников требуется специальный навигационный приемник с измененной прошивкой, т.к. в обычных приемниках эфемериды всех спутников заранее заложены, и сигналы псевдоспутников будут проигнорированы (не взяты в расчет навигационной задачи). Для приема сигналов псевдоспутников требует специальная прошивка приемника. Кроме того, сигнал псевдоспутников может передавать на иных частотах, нежели сигналы глобальной радионавигационной системы, поэтому переработке подлежит также радиочасть приемника.

Достоинством применения локальных радионавигационных полей является:

- более высокая точность;
- более высокий уровень сигнала (сложность подавления);
- возможность использования внутри помещений (при соблюдении ряда условий).

Недостатки:

- сложность обеспечения прямой видимости всех псевдоспутников;
- сложность обеспечения большого разброса по высоте установки, без которого точность определения высоты потребителя существенно падает (речь не идет об использовании псевдоспутника в режиме наземного дополнения);
- потребность переработки аппаратуры потребителя;
- для псевдоспутников не утвержден частотный диапазон.

4.1.6. Системы координат глобальных навигационных систем

Системы глобальной навигации формируют решение навигационной задачи в декартовой *геоцентрической системе координат*. В этой же системе координат работают навигационные спутники.

Геоцентрическая система координат связана с Землей (Рис. 50). Центр системы координат находится в центре масс Земли. Ось Z направлена на Северный полюс. Ось X в координаты 0º северной широты, и 0º восточной долготы (территория Гвинейского залива). Ось X дополняет систему координат до правой тройки векторов.

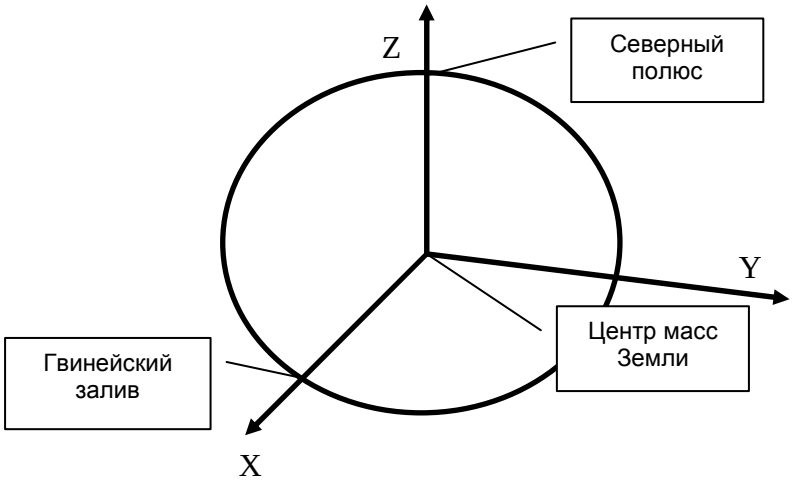

Рис. 50 Геоцентрическая система координат

На выходе навигационный приемник по умолчанию формирует географические координаты (долготу, широту и высоту над уровнем опорного эллипсоида). Причем GPS выдает долготу, широту и высоту в устаревшей системе координат *WGS-84*, а ГЛОНАСС – в более уточненной системе координат *ПЗ-90.02*.

Следует отметить, что Земля представляет собой геоид, который для простоты описывается эллипсоидом. Эллипсоид описывает уровень моря с погрешностью не хуже 200 метров. Системы координат WGS-84 и ПЗ-90.02 отличаются параметрами опорного эллипсоида.

Система координат WGS-84 принятая в 1984 году, предполагает, что радиус Земли по экватору составляет 6 378 137 м, а коэффициент сжатия 1/298.257223563.

Система координат ПЗ-90.02, принятая в 90 году, и уточненная в 2005 году, предполагает радиус Земли по экватору составляет 6 378 136 м, а коэффициент сжатия 1/298.25784.

Также в этих системах координат имеются небольшие разногласия местоположения центра масс Земли.

Разница в параметрах опорного эллипсоида для систем координат WGS-84 и ПЗ-90.02 при пересчете из геоцентрической системы координат в географические координаты дает разные результаты. Таким образом, точка, заданная долготой и широтой в системе координат WGS-84, на широтах Москвы может на 2-3 метра отличаться от точки, заданной той же долготой и широтой в системе координат ПЗ-90.02.

Для навигации мобильных роботов в радиусе нескольких десятков километров удобно пренебречь радиусом скругления Земли и считать землю плоской. При этом формируется система координат на этой базовой плоскости. Для этого необходимо выбрать за начало координат некоторую базовую точку *B*. Через точку *B* провести по касательной к земному эллипсоиду базовую плоскость (плоскость земли). В данном плоскости будут располагаться оси $X_п$ и $Y_п$ новой системы координат. Ось $Y_п$ удобно направить на север, ось $X_п$ – на восток, а ось высоты $H_п$ вверх от центра Земли (Рис. 51). Подобная система координат называется «*система координат на базовой плоскости*».

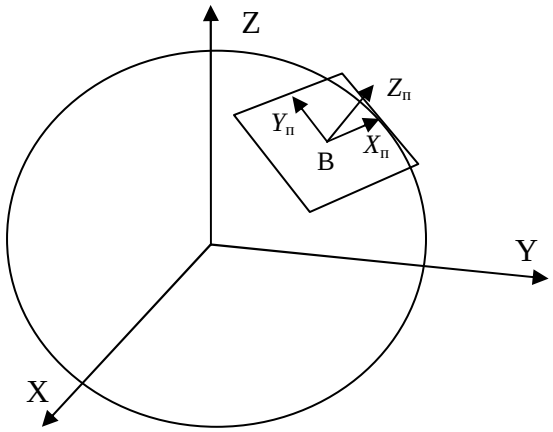

Рис. 51 Система координат на базовой плоскости

4.1.7. Оборудование для приема радионавигационных сигналов

Для приема сигналов радионавигационных сигналов существует множество готовых приемников, бескорпусных модулей, отдельных микросхем, а также автомобильных навигаторов.

Некоторые приемники работают исключительно по сигналам GPS, а некоторые работают по GPS и ГЛОНАСС одновременно, лишь некоторые из них работают исключительно по сигналам ГЛОНАСС.

Почти все приемники имеют COM-порт для связи с персональным компьютером. Некоторые из них реализуют виртуальный COM-порт по USB (см. микросхема FT323RL). Эти приемники поддерживают стандартный протокол BINARY или NMEA. Причем, в большинстве

случаев приемник автоматически определяют формат передаваемых на него команд и используют соответствующий протокол.

Некоторые приемники не имеют собственной антенны и требуют подключения внешней. В этом случае необходимо подключить к приемнику антенну с полусферической диаграммой направленности на рабочий диапазон 1500 МГц.

Хорошо зарекомендовали себя активные антенны, имеющие встроенный усилитель мощности. В отличие от пассивных антенн, имеющих длинный кабель, выступающий в роли дополнительной паразитной антенны, в активной антенне расстояние от фазового центра антенны до усилителя минимально. Поэтому активные антенны позволяют более точно измерять местоположение, кроме того, из-за наличия внешнего усилителя антенна принимает больше спутников.

Питание активной антенны осуществляется по тому же кабелю, что и передается сигнал (постоянное напряжение в кабеле никак не влияет на передачу по нему высокочастотного сигнала). Для подключения активной антенны существуют специальные инжекторы питания. Некоторые профессиональные приемники уже имеют вход для подключения активной антенны. Однако при подключении активной антенны пользователь должен быть уверен в том, что приемник имеет развязку по питанию высокочастотной части.

Активные антенны имеют различное напряжение питания и различную потребляемую мощность. Поэтому пользователю следует обращать внимание на этот аспект при выборе активной антенны и ее источника питания. Также следует отметить, что к выходу активной антенны нельзя подключать пассивную антенну, т.к. при этом возникает короткое замыкание, которое может привести к выходу из строя как источника питания, так и самой антенны.

4.1.8. Использование систем глобальной навигации в Dyn-Soft RobSim 5

Для использования систем глобальной навигации в Dyn-Soft RobSim 5 необходимо установить приемник GPS/ГЛОНАСС (система глобальной навигации), а также антенну (если требуется) на диапазон 1500 МГц (Рис. 52).

При выборе конкретной модели приемника следует следить за напряжением его питания, интерфейсом связи и обеспечением требуемой точности позиционирования.

При выборе антенны также следует руководствоваться диапазоном частот, на который рассчитана антенна. Для GPS и ГЛОНАСС диапазон должен быть 1500 МГц. Если используется пассивная антенна, то рекомендуется использовать антенну «ШАЙБА» отечественного производства.

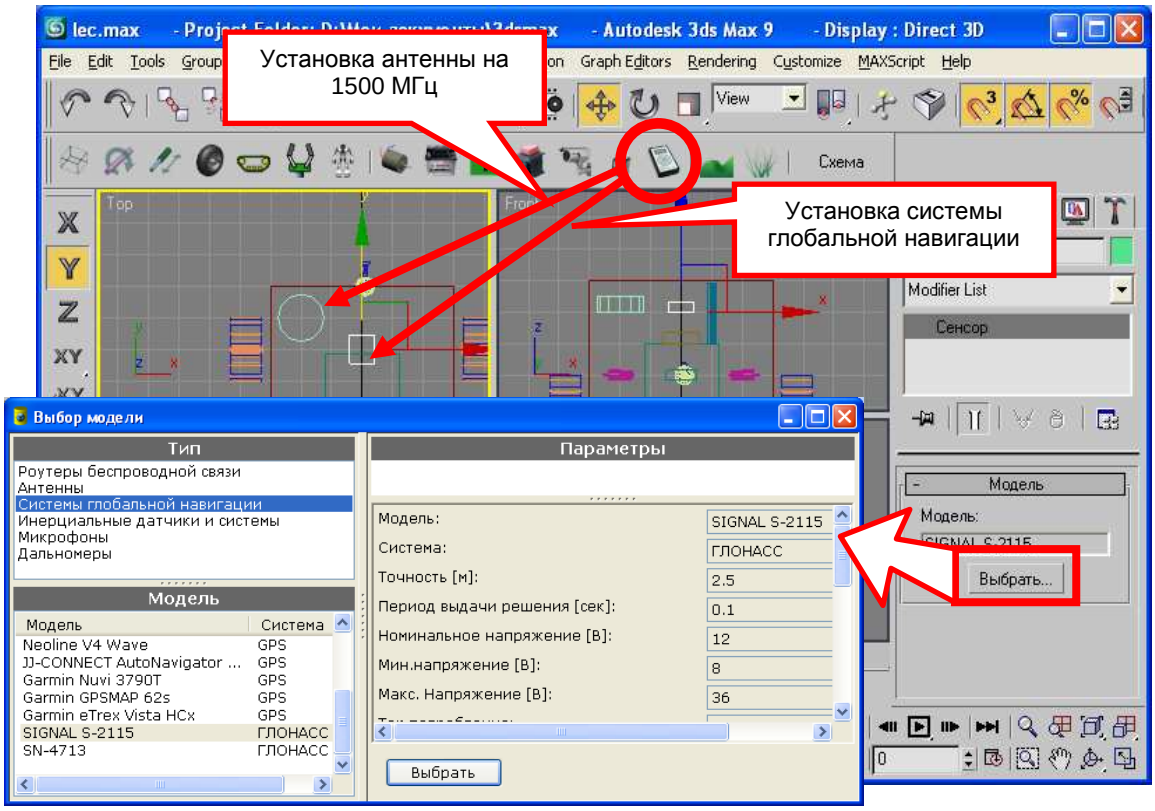

Рис. 52 Иллюстрация установки навигационного приемника в 3D Studio MAX

Используя редактор схем и подключений (кнопка «Схема») антенну следует подключить на вход навигационного приемника, а сам навигационный приемник подключить к COM-порту или USB-порту бортовой ЭВМ (в зависимости от используемого интерфейса) (Рис. 54). Если приемник не использует питание от USB, то для него также следует обеспечить питание.

Для работы с системами радионавигации в редакторе структурных схем программного обеспечения Dyn-Soft RobSim 5 имеется целый ряд блоков (Рис. 53).

Для управления навигационным приемником в Dyn-Soft RobSim 5 следует использовать блок «Драйвер навигационного приемника». При этом COM-порт (или USB-порт) следует настроить на прием навигационных данных по каналу 0, а передачу управления навигационным приемником по каналу 1 (Рис. 55). Для COM-порта его следует настроить на скорость передачи данных 9600 бод, 1 стоповый бит, 8 бит данных без контроля четности, асинхронный режим приема-передачи данных.

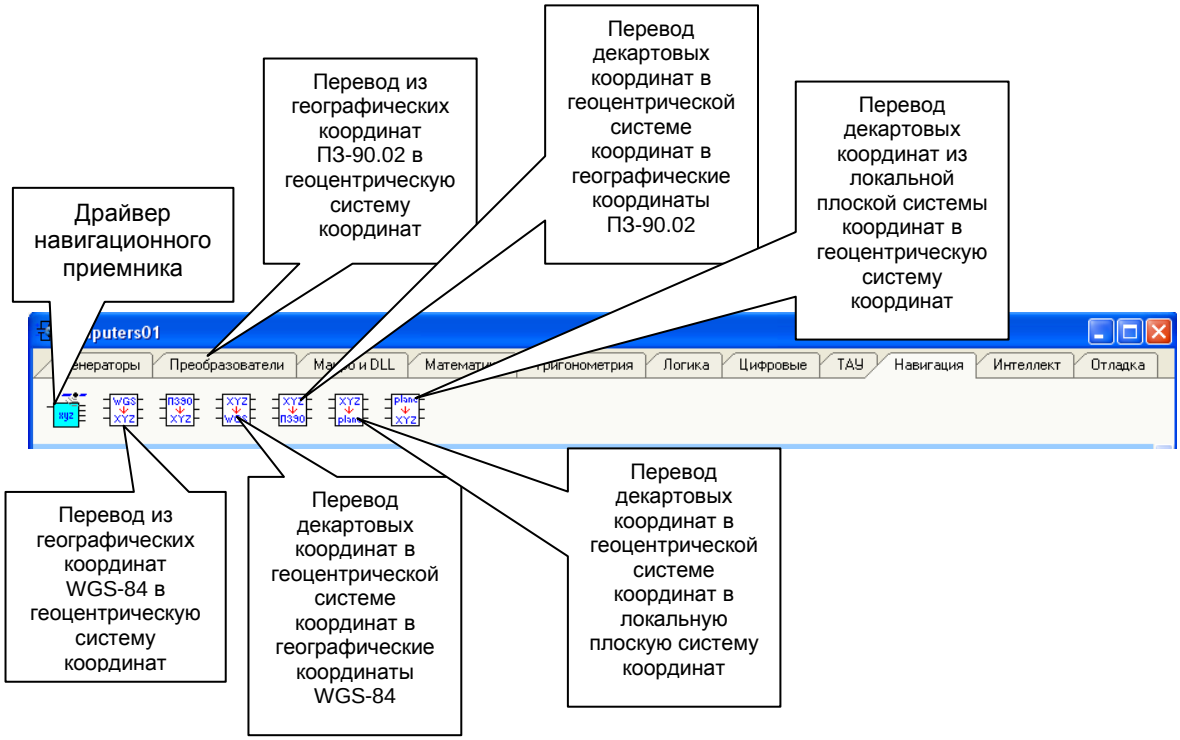

Рис. 53 Внешний вид закладки «Навигация» в палитре редактора структурных схем программного обеспечения бортовой ЭВМ

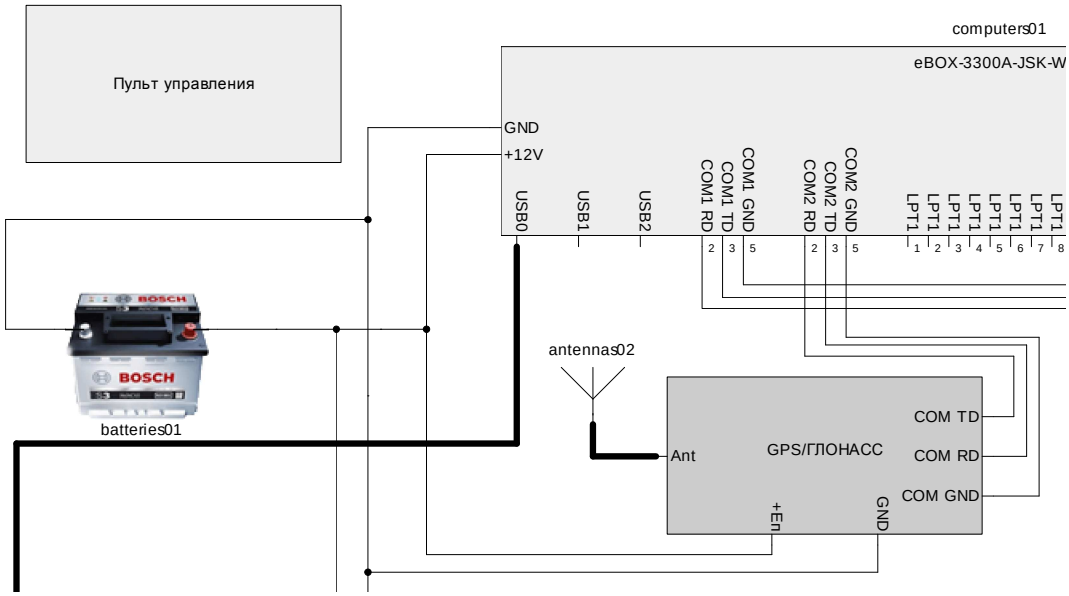

Рис. 54 Пример подключение навигационного приемника к бортовой ЭВМ

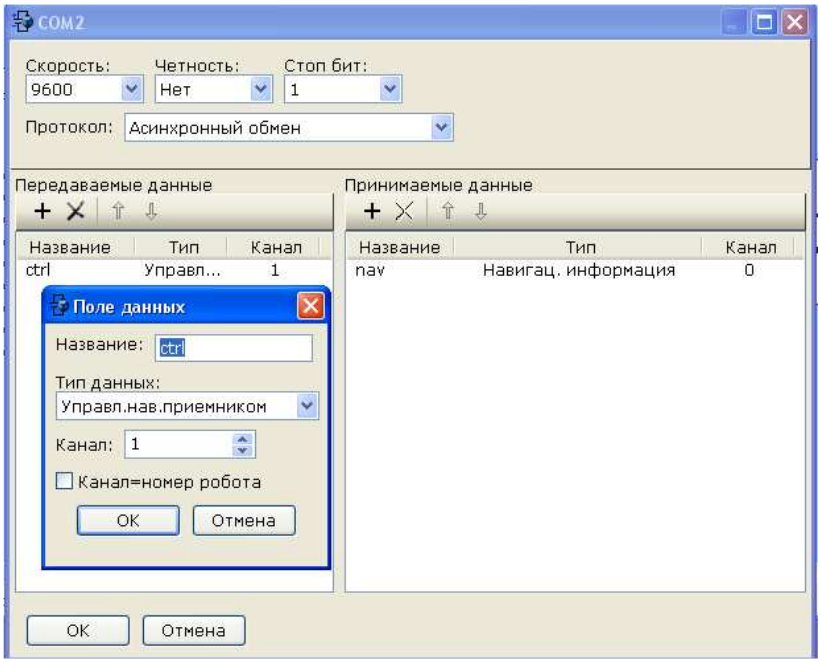

Рис. 55 Иллюстрация настройки канала связи для связи с навигационным приемником

Схема подключения к драйверу навигационного приемника должна быть такой, как приведена на Рис. 56.

В свойствах драйвера навигационного приемника можно выбрать систему координат, в которой драйвер будет выдавать свои показания. Можно выбрать WGS-84 (по умолчанию), ПЗ-90.02 или геоцентрическую систему координат.

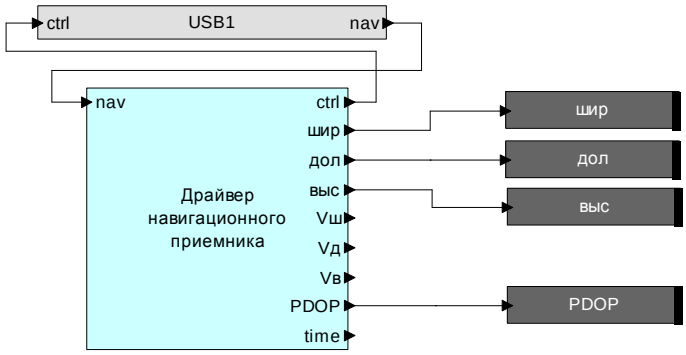

Рис. 56 Схема структуры программного обеспечения для бортовой ЭВМ, иллюстрирующая подключение навигационного приемника

Следует отметить, что драйвер навигационного приемника выдает показания долготы и широты в радианах, а высоту над уровнем опорного эллипсоида – в метрах. Причем для серверной широты значения положительные, а для южной – отрицательные. Для восточной долготы показания положительные, а для западной – отрицательное.

Также с драйвера навигационного приемника можно получить значение скорости, разложенное по осям долготы, широты и высоты (в случае, если навигационный приемник формирует на выходе географические координаты), если значение скорости, разложенное по осям геоцентрической системы координат (если в параметрах драйвера навигационного приемника выбрана геоцентрическая система координат).

Для проведения экспериментальных исследований работы навигационного приемника в Dyn-Soft RobSim 5 следует использовать только сцены, в которых задана географическая привязка (см. раздел «Разработка виртуальных сцен» в учебном пособии «Проектирование роботов и робототехнических систем, часть I»). Например, сцену «Polygon».

Процесс проведения экспериментальных исследований показан на Рис. 57.

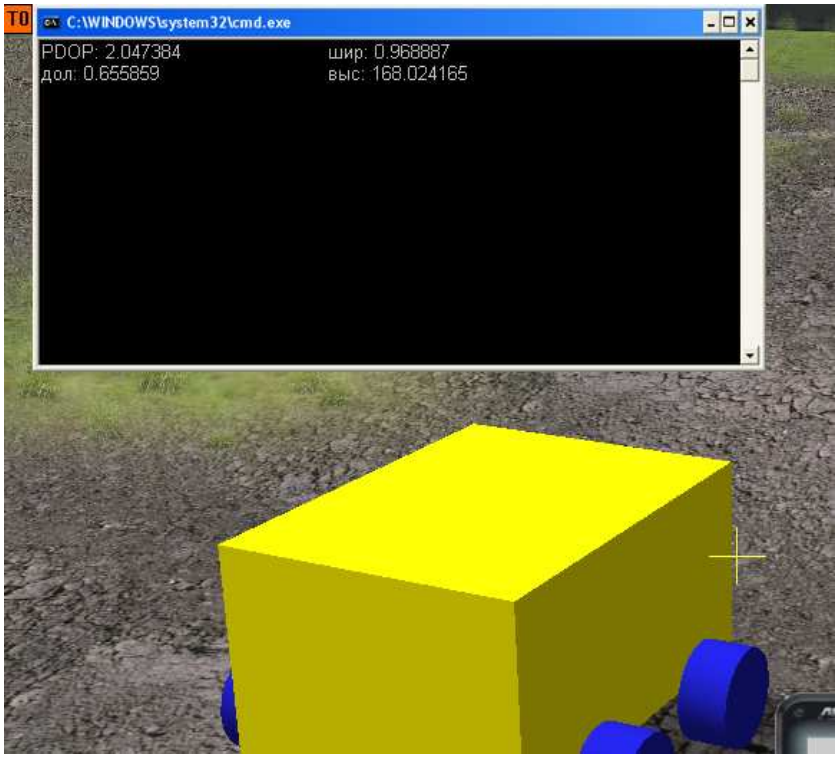

Рис. 57 Иллюстрация процесса проведения экспериментальных исследований работы навигационного приемника в Dyn-Soft RobSim 5

4.1.9. Радионавигация мобильного робота в локальной декартовой системе координат в Dyn-Soft RobSim 5

Для навигации мобильного робота удобно использовать локальную систему координат, центр которой расположен в центре рабочей сцены, ось *Y* направлена на север, *X* на восток, а вертикальная ось *H* направлена вверх.

Для пересчета показаний навигационного приемника в локальную систему координат следует перевести географические координаты в

геоцентрическую систему координат, а затем в координаты на базовой плоскости (Рис. 51).

Как видно из рисунка для такого перехода необходимо знать базовую точку *B*. Для фиксации базовой точки предлагается использовать схему с RS-триггером, срабатывающим, как только геометрический фактор (PDOP) будет приемлемый (меньше 3), и трех D-триггеров, которые защелкнут показания координат *X,Y,Z* в геоцентрической системе координат в момент срабатывания RS-триггера. Также на схему будут установлены 3-регистра, запись в которые будет разрешена только тогда, когда геометрический фактор будет меньше 3. В противном случае регистры будут выдавать последнее запомненное значение (Рис. 58). Важно отметить, что результат будет получиться в метрах.

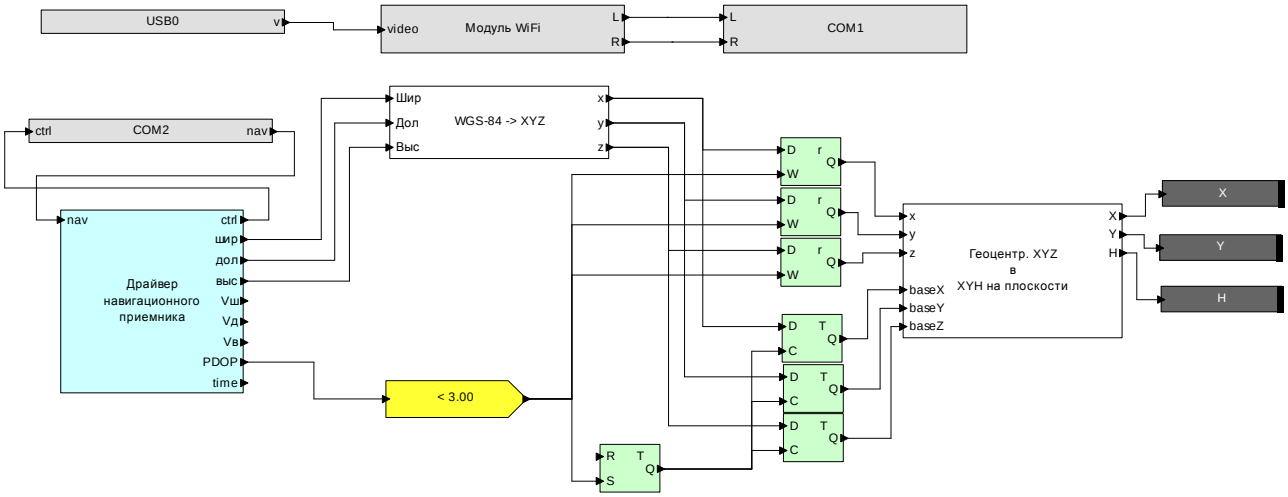

Рис. 58 Структура программного обеспечения для бортовой ЭВМ для перевода показаний навигационного приемника в локальную систему координат на базовой плоскости

Таким образом, мобильный робот изначально находится в начале системы координат, и находится там до тех пор, пока приемник не начнет выдавать достоверные координаты (PDOP не станет меньше 3). Как только геометрический фактор станет приемлемым, D-триггеры запомнят показания и будут их использовать как координаты базовой точки. Регистры же также будут пропускать через себя только те координаты, геометрический фактор которых меньше 3.

Для навигации мобильных роботов также важно знать угол ориентации робота. Системы глобальной навигации напрямую не позволяет определить этот угол, поэтому рекомендуется использовать другие системы определения угла ориентации.

Однако, зная вектор скорости, можно вычислить направление движения робота. Если предположить, что робот может двигаться только

вперед, то направление его скорости совпадает с ориентацией. Но определить ориентацию можно только в движении.

Удобно, что драйвер навигационного приемника возвращает вектор скорости, разложенный по долготе, широте и высоте. Угол между составляющими скорости по широте и долготе дают требуемый угол ориентации. Вычислить этот угол можно с помощью функции arctg2. Следует также отметить, что вычислять угол ориентации можно только, если длина вектора скорости больше, некоторой минимальной скорости, в противном случае флюктуация вектора скорости будет давать очень большую погрешность определения угла ориентации. Для этого на схему необходимо установить регистр, запись в который будет разрешена только при возникновении благоприятных условий определения направления движения. Также, следует также снизить уровень шумов по скорости, возвращаемый приемником. Для этого составляющие вектора скорости имеет смысл пропустить через фильтр низких частот (апериодические звенья с постоянной времени 0.1 – 0.2 сек).

Структурная схема программного обеспечения для определения угла ориентации робота по сигналам спутниковой навигации приведена на Рис. 59. Данная схема работает совместно со схемой на Рис. 58.

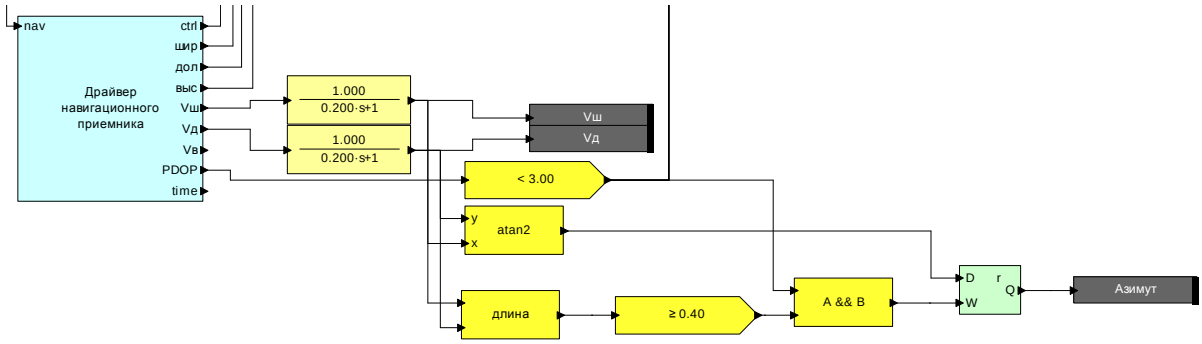

Рис. 59 Структура программного обеспечения бортовой ЭВМ для определения угла ориентации мобильного робота по сигналам спутниковой навигации

Таким образом, был показан пример установки, подключения и использования системы глобальной навигации в среде Dyn-Soft RobSim 5 для определения местоположения робота и его ориентации. Однако, как было показано выше, система глобальной навигации не обеспечивает высокой точности позиционирования, поэтому применима только для выведения робота на открытой местности в район с радиусом 10-15 метров. Внутри помещений система глобальной навигации не работает.

4.2. Инерциальные навигационные системы

4.2.1. Введение

Инерциальная навигационная система определяет свое местоположение и ориентацию на основе гироскопических датчиков и акселерометров. В некоторых случаях инерциальная навигационная система может быть дооснащена магнитометром (датчиком магнитного курса), и барометрическим датчиком высоты.

Линейные перемещения в инерциальной навигационной системе измеряются с помощью трех акселерометров (датчиков ускорения). Однако вектор ускорения измеряется в локальной системе координат прибора. Вектор ускорения необходимо перевести в мировую систему координат. Для этого необходимо знать абсолютную ориентацию прибора, которая измеряется с помощью гироскопических датчиков.

После перевода вектора ускорения в мировую систему координат, ускорения интегрируются. В результате формируется вектор скорости в мировой системе координат. Путем интегрирования вектора скорости формируется вектор положения.

Несложно заметить, что двойное интегрирование, применяемое в приборе, дает большую ошибку. Поэтому точность инерциальной системы навигации весьма мала, а ошибка растет с течением времени.

В связи с этим инерциальные системы навигации применяют только в комплексе с другими навигационными системами, позволяющими периодически сбрасывать ошибку интегрирования.

Большинство инерциальных систем используют микромеханические гироскопические датчики, которые измеряют угловую скорость по трем осям в локальной системе координат. Путем перевода вектора вращения в мировую систему координат с последующей интеграцией можно получить углы ориентации системы относительно начальной ориентации. Однако, интегрирование в данном случае также копит ошибку и углы ориентации со временем «уплывают».

В некоторых инерциальных датчиках используется магнитометры, измеряющие магнитное поле земли (по аналогии со стрелочным компасом). Совместная работа с данным датчиком позволяет сбрасывать ошибку интегрирования ориентации. Однако данный датчик очень чувствителен к внешним магнитным полям и большому скоплению металлических объектов вокруг датчика. Поэтому определение угла ориентации только по магнитометру не является панацеей.

Также для помощи инерциальной системе навигации используют барометрические высотомеры. Однако точность измерения высоты по данному датчику порядка 1-2 метра.

Инерциальные системы с механическими гироскопами используются для создания систем навигации самолетов, ракет и кораблей во всем мире.

В ряде случаев инерциальные системы используют не столько для навигации, сколько для стабилизации движения. Например, с помощью микромеханического гироскопа стабилизирую вращение задней балки вертолетов, в т.ч. на радиомоделях вертолетов.

Подробно о каждом виде датчиков и о готовых инерциальных системах речь пойдет в следующих главах.

4.2.2. Механические гироскопы

Под механическими гироскопами обычно понимают трехстепенной роторный гироскоп.

Принцип действия гироскопа основан на возникновении прецессии быстро вращающегося ротора (Рис. 60). Если быстро раскрутить массивный ротор, подвешенный на трехосевом карданном подвесе, и направить его, например, на звезду, то, несмотря даже на вращение Земли ось ротора будет указывать на выбранное направление.

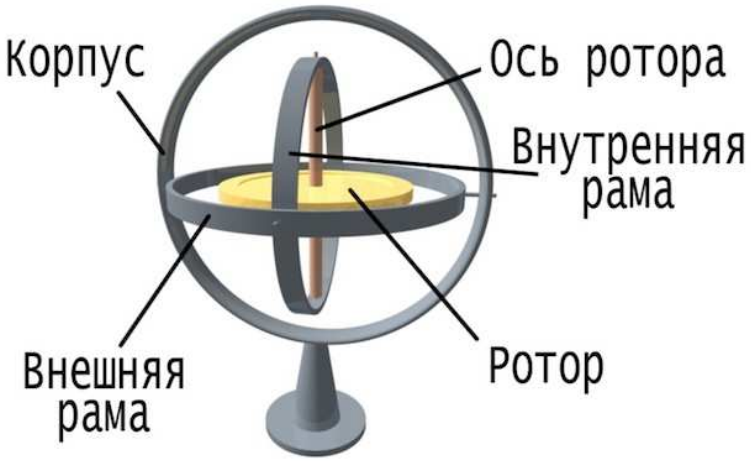

Рис. 60 Схема конструкции механического гироскопа

Поэтому, если измерять угол ориентации внешней рамы относительно корпуса, а также угол ориентации внутренней рамы относительно внешней рамы, можно определить два угла ориентации прибора относительно начальной ориентации. Причем с достаточно большой точностью. Важно, чтобы скорость вращения корпуса была на несколько порядков ниже скорости вращения ротора.

Для поддержания постоянной скорости вращения ротора используется электропривод.

Основным источником погрешности механического гироскопа, является наличие сил трения в осях карданного подвеса. Силы трения с течением времени изменяют углы ориентации ротора, поэтому со временем в механических гироскопах накапливается ошибка. Но данная ошибка на несколько порядков ниже, чем в аналогичных микромеханических гироскопах.

Для борьбы с данной погрешностью пытаются добиться большего доминирования кориолисовых сил над силами трения. Для этого увеличивают скорость вращения ротора и увеличивают его массу.

Поэтому масса и габаритные размеры механических гироскопов достаточно велики (Рис. 61). Известно, что гироскопы для атомных подводных лодок, которые месяцами должны находиться в автономном плавании, раскручивают еще на заводе-изготовителе, и включенными кранами монтируют в корпус лодки.

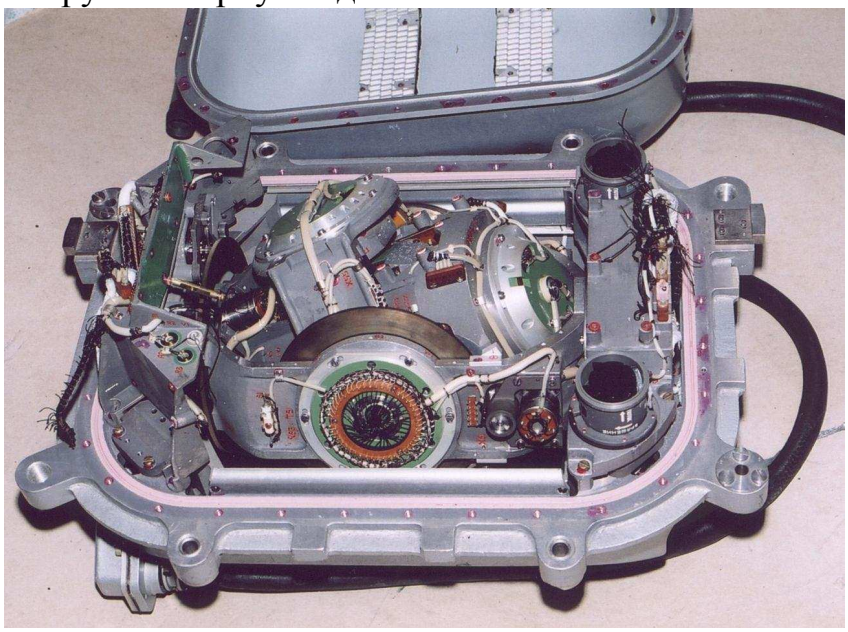

Рис. 61 Внешний вид трехстепенного роторного гироскопа ПГ-11-2

Следует отметить, что высокие обороты вращения ротора механического гироскопа создают достаточно неприятный и громкий звук. Так, например, пуск ракет воздушного базирования связан с раскруткой их внутреннего гироскопа, что создает не меньше шума, чем реактивный двигатель ракеты.

В робототехнике механические гироскопы практически не применяют из-за их высокой стоимости, габаритов и массы.

4.2.3. Микромеханические гироскопы. Датчики угловых скоростей (ДУС)

Датчики угловых скоростей (ДУС) или микромеханические гироскопы относятся к вибрационным гироскопам.

Принцип работы любого гироскопа основан на явлении прецессии. Из курса механики известно, что явление прецессии заключается в преобразовании вектора силы, приложенного перпендикулярно плоскости вращения, в вектор, перпендикулярный исходному и оси вращения. Явление наблюдается при любых механических движениях с ненулевой угловой скоростью. Значит, это не обязательно вращение, как в механических гироскопах. При изгибном колебании также имеется знакопеременная угловая скорость. Данный фактор и используется в гироскопа вибрационного типа.

Используя принцип действия вибрационных гироскопов, появилась возможность изготовлять гироскопы по технологии MEMS (микромеханика). По данной технологии можно изготавливать гироскопы, размером менее 1 мм (Рис. 62).

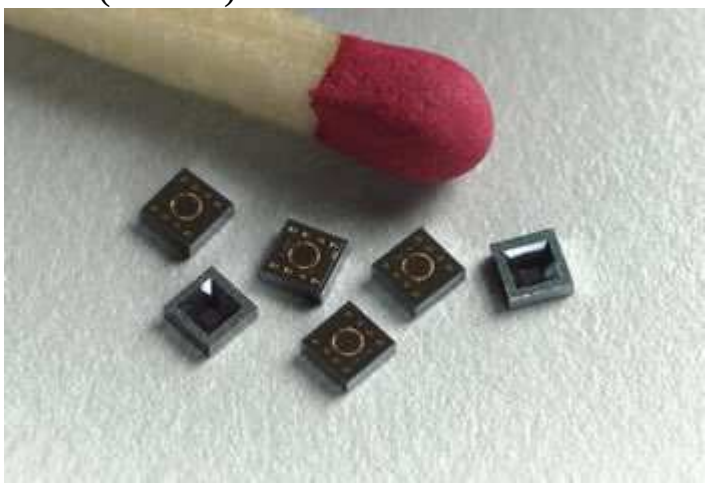

Рис. 62 Микромеханические гироскопы

В настоящее время гироскопы отличаются по способу снятия колебаний в перпендикулярной оси. Различают пьезоэлектрические, индуктивные и емкостные гироскопические датчики. Несмотря на различные способы снятия первичной информации, интерфейс подключения и характер формируемых сигналов датчиков не отличается.

В отличие от механических гироскопов, микромеханические гироскопы на выходе не формируют сигналы, пропорциональные углам отклонения от первоначального направления. Вместо этого на выходе микромеханических гироскопов – величина пропорциональная угловой скорости вращения.

Для полноценной замены механического гироскопа необходимо, как минимум, три микромеханических датчика угловых скоростей, расположенных по трем осям прибора, а также специальное вычислительное устройство, которое будет производить интегрирование скоростей для получения трех углов ориентации.

Следует отметить, что датчики угловых скоростей измеряют скорость вращения относительно осей локальной системы координат

прибора, а интегрировать их следует относительно осей мировой (начальной) системы координат.

Перед интегрированием сигналов с микромеханических гироскопов рекомендуется применять фильтры низких частот или скользящие средние для усреднения показаний ДУС на некотором интервале времени. В этом случае резко уменьшаются шумы датчиков, и уменьшается погрешность. Но наличие фильтров снижают диапазон угловых ускорений, в котором может работать устройство.

Важно обращать внимание на диапазон скоростей, измеряемых с помощью ДУС. Этот диапазон крайне важен для процесса интегрирования. С одной стороны, если датчик имеет большой диапазон угловых скоростей, следовательно, он имеет большую цену младшего разряда (даже если у датчика аналоговый выход). Т.е. датчик имеет низкую точность измерения угловой скорости. Такой датчик нечувствителен к медленному вращению. А медленным вращением можно развернуть устройство на 180^{o}.

С другой стороны, если датчик имеет высокую точность, то его диапазон измерений, скорее всего, небольшой. Т.е. достаточно резко сотрясти устройство (например, при соударении), как в системе появляется кратковременные высокие угловые скорости. Датчик войдет в насыщение, и измеряемая угловая скорость не будет совпадать с реальной. Процесс интегрирования скорости получит неверные исходные данные, что приведет к большому рассогласованию между реальной и измеренной ориентацией прибора.

Несложно заметить, что за счет наличия интегрирования, система может накапливать погрешность. Поэтому система навигации на основе ДУС следует иметь механизм сброса ошибки интегрирования.

ДУС обычно имеет либо аналоговый выход скорости, либо цифровой I2C или SPI.

На рынке сейчас имеется несколько микросхем цифровых гироскопических датчиков. Одна из них L3G4200D, представляющая собой трехосевой гироскоп. На Рис. 63 изображена электронная плата на базе данной микросхемы.

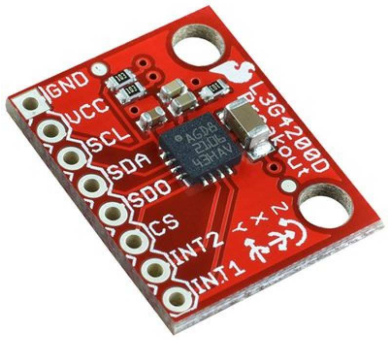

Рис. 63 Электронная плата на базе гироскопа L3G4200D

4.2.4. Акселерометры

Акселерометр – датчик ускорения.

Все современные акселерометры изготавливают по технологии MEMS (микромеханика).

Принцип действия акселерометра достаточно прост: на подпружиненном креплении устанавливает грузик. При возникновении ускорения вдоль измеряемой оси грузик отклоняется от своего нормального положения, причем тем больше, чем выше ускорение. В микромеханических акселерометрах различают несколько способов определения положения этого грузика: емкостной, пьезоэлектрический или индуктивный.

В микромеханических гироскопах пытаются всяческим способом компенсировать колебания грузика. Тем не менее, полностью компенсировать шум акселерометра невозможно. Поэтому выходной сигнал подергают фильтрации. Применяют скользящую среднюю или фильтр низких частот.

Не следует забывать, что акселерометр изменяет не только ускорение движения объекта, но и ускорение свободного падения. Поэтому при наклоне акселерометра, ускорение свободного падения проецируется на ось акселерометра. Поэтому довольно часто в таких устройствах, как телефоны и планшетные компьютеры, акселерометр используется в качестве датчика наклона.

Для создания полноценной навигационной системы необходимо применять систему из трех акселерометров по каждой оси, совместно с трехосевым гироскопом.

При начальной калибровке устройства сразу после включения, когда устройство заведомо находится в состоянии покоя, система управления должна измерить вектор ускорения свободного падения. При последующей работы устройства необходимо показания трехосевого акселерометра с учетом ориентации устройства перевести в мировую систему координат, вычесть начальный вектор ускорения свободного падения, после чего начать интегрировать. Тем самым первое интегрирование дает вектор скоростей, второе – вектор положения в пространстве относительно начальной точки.

Выходом акселерометра может быть, как аналоговый сигнал, так и цифровой: ШИМ, SPI или I2C. Обычно ускорение измеряют в $g = 9.8$ м/с^2.

При использовании акселерометров следует учитывать диапазон их измерений. Если диапазон измерений большой, то датчик нечувствителен как к малым ускорениям, так и имеет высокую погрешность измерения. Т.е., применяя невысокое ускорение, можно разогнать устройство до некоторой скорости, а датчик ничего не почувствует. Т.о., после первого интегрирования устройство должно показывать скорость, но она будет нулевой. Интегрирование нулевой скорости также получит нулевое

приращение положения. Т.е. устройство двигается, а датчики показывают, что устройство стоит на месте.

Если диапазон измерений маленький, то кратковременные высокие ускорения, возникающие, например, при соударениях, вводят акселерометры на насыщение. В результате акселерометр показывает заниженное ускорение, после первого интегрирования формируется неправильная скорость, а после второго интегрирования появляется большая ошибка позиционирования.

Поэтому использование инерциальной системы навигации без механизма сброса ошибки интегрирования невозможно.

Обычно выпускают двух или трехосевые акселерометры. Например, микросхема ADXL203CE (Рис. 64).

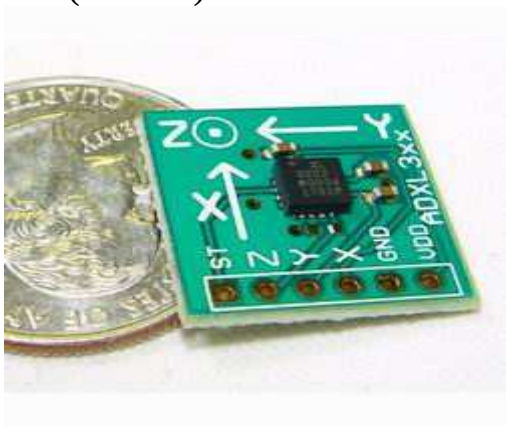

Рис. 64 Пример платы с трехосевым акселерометром на базе микросхемы ADXL203CE

4.2.5. Магнитометры

Магнитометр – датчик магнитного поля Земли. Подобно компасу, трехосевой магнитометр показывает вектор направления на север, точнее на магнитный полюс Земли.

Магнитомер собирают на базе трех магниторезистривных датчиков, например, на базе микросхемы KMZ51. Обычно микросхема имеет аналоговый выход.

Следует отметить, что магнитометр крайне чувствителен к наличию магнитных полей или металлических объектов вокруг устройства. Так, например, микроподводная лодка, запущенная в бассейне, направление на север определяла вдоль металлического поручня, протянутого по периметру бассейна.

В фильме «Пятнадцатилетний капитан», Кок Негоро, оказавшийся агентом работорговцев, обманом меняет курс корабля с помощью железного топора, подложенного под компас, и вместо Боливии (Южной Америки) судно «Пилигрим» приходит к берегам Анголы в Африке.

Поэтому полностью доверять показаниям магнитометра нельзя.

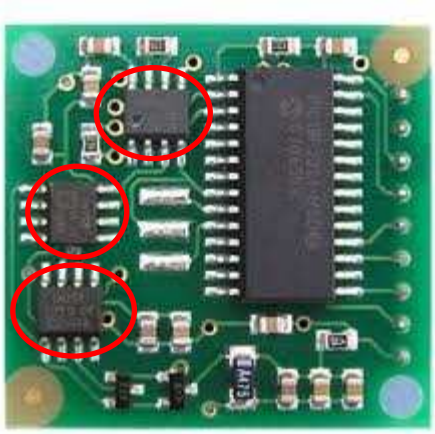

Рис. 65 Плата магнитного компаса (IE-CMPS-03) на базе трех микросхем KMZ51

4.2.6. Барометрические датчики высоты

Известно, что с ростом высоты падает атмосферное давление. Падение давления почти линейно связано с высотой. Данный эффект используют в барометрических высотомерах. Для использования таких высотомеров следует определить и запомнить показание барометра в точке с известной высотой (например, взять текущую высоту за нуль). После этого вычитать из запомненного давления текущие показания барометра, которые через константу пересчитывается в высоту.

Для более качественного измерения высоты следует производить температурную компенсацию, поэтому обычно барометрические датчики объединены с датчиком температуры. В документации на такие микросхемы обычно приводится формула температурной компенсации. Примером таких датчиков является микросхема MPL115A (Рис. 66).

Рис. 66 Внешний вид микросхемы MPL115A
(барометр и датчик температуры в одном корпусе)

Данная микросхема имеет интерфейс I2C.

Следует отметить, что определить высоту по барометрическому высотомеру точнее 1-2 метров невозможно. Сказываются как погрешность измерения давления, так и его флюктуации.

4.2.7. Датчики инерциальных навигационных систем

На рынке имеется довольно широкий ассортимент готовых плат, включающих в свой состав трехосевые акселерометры, гироскопы, датчики магнитного курса и барометры в различных сочетаниях.

В большинстве плат, позиционирующих себя, как инерциальные системы, уже имеется вычислитель, производящий все необходимые интегрирования показания датчиков. На выходе таких устройств можно получить как сырые показания инерциальных датчиков, так и готовый результат в виде положения и ориентации устройства. Например, плата CHR-UM6-LT, включающая в свой состав трехосевой гироскоп, трехосевой акселерометр и трехосевой магнитометр. Удобство применения данной платы является напряжение питания 5В, интерфейс RS-232 TTL (UART) или SPI.

4.2.8. Интерфейсы датчиков инерциальных навигационных систем

Различные датчики навигационных систем имеют разные интерфейсы. Наиболее распространены:

- Аналоговый выход – выход с датчика или блока датчиков представляет собой аналоговый сигнал (по каждому из каналов). Уровень напряжения по данному выходу прямо пропорционален измеряемой датчиком величине. Нулевому показанию датчика обычно соответствует средняя величина рабочего напряжения устройства (которое, обычно составляет 3.3В). Относительно данного уровня напряжения формируется как положительные, так и отрицательные показания. Разработчику важно ознакомиться с документацией на датчик, в которой должен быть обязательно указан коэффициент пропорциональности между уровнем напряжения и измеряемой величиной. Для оцифровки показаний таких датчиков рекомендуется использовать АЦП, например, встроенный АЦП микроконтроллеров AVR.
- ШИМ – выход с датчика или блока датчиков представляет собой ШИМ-последовательность (по каждому из каналов). Коэффициент заполнения ШИМ пропорционален измеряемой величине. Нулевому показанию датчика соответствует ШИМ с коэффициентом заполнения 50%. Относительно данного

коэффициента заполнения формируются как положительные, так и отрицательные показания. Следует отметить, что устройство с ШИМ-выходом более предпочтительно, чем аналоговый выход, т.к. значительно меньше подвержен влиянию паразитных аналоговых шумов в кабеле связи с управляющим устройством. Приемное устройство рекомендуется изготавливать на базе контроллеров AVR, и использовать в них анализатор ШИМ-последовательности.

- RS-232 – устройство управляется по RS-232. Обычно протокол связи с устройством описан в документации на устройство. В Dyn-Soft RobSim 5 имеется универсальный драйвер для всех типов инерциальных датчиков, позволяющий получать показания каждого канала.
- RS-232 TTL, TTL или UART – некоторые производители по разному называют данный вид интерфейса, представляющего собой сигнал, известный в Dyn-Soft RobSim 5 под названием UART. Единственное, следует отметить, что обычно инерциальные датчики формируют UART с уровнем напряжения 3.3В (а не 5В). Однако обычно входной сигнал UART толерантен к 5В. Это позволяет данное устройство без дополнительных согласующих устройств подключать к UART-интерфейсу с уровнем напряжений 5В. Протокол связи с устройством обычно содержится в документации на блок датчиков. В Dyn-Soft RobSim 5 имеется универсальный драйвер для всех датчиков инерциальных систем навигации.
- SPI – протокол SPI. Физический уровень данного протокола аналогичен UART. Подключить устройство рекомендуется к SPI-интерфейсу AVR параллельно программатору. В Dyn-Soft RobSim 5 имеется универсальный драйвер для всех датчиков инерциальных систем навигации, который поддерживает связь в т.ч. через SPI-интерфейс.
- I2C – протокол I2C. Для подключения можно использовать TWI-интерфейс микроконтроллеров AVR (аналог I2C в AVR). Линии связи следует подтянуть к $+E_{\text{п}}$ через резисторы 1кОм. В Dyn-Soft RobSim 5 имеется универсальный драйвер для всех датчиков инерциальных систем навигации, который поддерживает связь в т.ч. через TWI-интерфейс.

4.2.9. Интегрирование показаний инерциальных навигационных датчиков

Для реализации непосредственно навигации на основе показаний датчиков инерциальной системы необходимо показания датчиков специальным образом обработать.

Первичная фильтрация

Важным этапом обработки показаний навигационной системы является первичная фильтрация. Дело в том, что уровень шумов датчиков достаточно велик.

Для фильтрации обычно применяют алгоритм скользящих средних, который заключается в следующем: пусть на вход системы поступают «сырые» показания $x^{<raw>}$ датчика в виде дискретных измерений с постоянным периодом. Разработчик определяется с уровнем фильтрации, выбирая величину N – количество элементов фильтрации (рекомендуется использовать $N \geq 10$).

Из N элементов составляется буфер FIFO:

$$X_i := X_{i+1} \quad \text{для} \quad i = 0 \ldots N-2$$

$$X_{N-1} := x^{<raw>}$$

Здесь: X_i – буфер FIFO. Знаком «:=» означен оператор присвоения.

Выходное отфильтрованное значение «сырых» данных $y^{<raw>}$ рассчитывается, как среднее арифметическое FIFO-буфера:

$$y^{<raw>} = \frac{\sum_{i=0}^{N-1} X_i}{N}$$

Чем больше величина N, тем больше уровень фильтрации. Шумы системы уменьшаются, но и быстродействие датчика падает. Т.е. быстрые перемещения или вращения системе будут недоступны.

Почти тот же результат фильтрации можно получить, пропустив показания датчика через апериодическое звено (по сути, фильтр низких частот):

$$y^{<raw>} := y^{<raw>} + \frac{x^{<raw>} - y^{<raw>}}{N}$$

Калибровка

Важнейшим этапом работы системы является тщательная первичная калибровка каждой оси датчика. Для этого важно в состоянии покоя

запомнить средний уровень сигнала (после фильтрации) по каждой оси. Т.е. нулевой уровень «сырых» ускорений акселерометра:

$$\vec{a}_0^{<raw>} = \left(a_{x,0}^{<raw>}, a_{y,0}^{<raw>}, a_{z,0}^{<raw>}\right),$$

и угловых скоростей гироскопа:

$$\vec{\omega}_0^{<raw>} = \left(\omega_{x,0}^{<raw>}, \omega_{y,0}^{<raw>}, \omega_{z,0}^{<raw>}\right).$$

Важно отметить, что нулевой уровень для акселерометра следует измерять, отдельно создавая состояние покоя для каждой оси. В противном случае на калибровочное значение будет влиять ускорение свободного падения.

Измерение вектора ускорения свободного падения

Известно, что точное значение ускорения свободного падения в разных точках Земли различно. Оно зависит от широты и высоты над уровнем моря. Например, в Москве ускорение свободного падения составляет 9,81523 м/с^2, в то время, как на полюсах оно равно 9,83 м/с^2, а на экваторе 9,78 м/с^2. Также имеются точки на Земле с гравитационными аномалиями. С увеличением высоты ускорение свободного падения также незначительно уменьшается.

Поэтому точное «сырое» значение вектора ускорения свободного падения $\vec{g}_0^{<raw>}$ относительно робота следует вычислять перед началом работы, тогда, когда робот находится на ровной площадке в состоянии покоя. Ненулевое направление вектора свободного падения учет неровности установки блока датчиков на робота.

$$\vec{g}_0^{<raw>} = \vec{a}^{<raw>} - \vec{a}_0^{<raw>}$$

Где: $\vec{a}^{<raw>}$ – измеряемое датчиком «сырой» вектор ускорений; $\vec{a}_0^{<raw>}$ – калибровочное значение датчика.

Матрицы преобразования координат

Несложно заметить, что инерциальная навигационная система измеряет показания датчиков относительно своей локальной системы координат (Рис. 67). Положение и ориентацию устройства следует формировать в мировой системе координат. Также именно в мировой системе координат следует производить описанную ниже процедуру интегрирования показаний датчиков.

Поэтому в вычислителе инерциальной навигационной системе неизбежно применение операций преобразования координат.

Преобразование координат удобно проводить с использованием матриц преобразования 4x4, или отдельно в виде матрицы поворота 3x3 и

вектора положения *x,y,z*. Для ускорения расчетов рекомендуется использовать последний вариант.

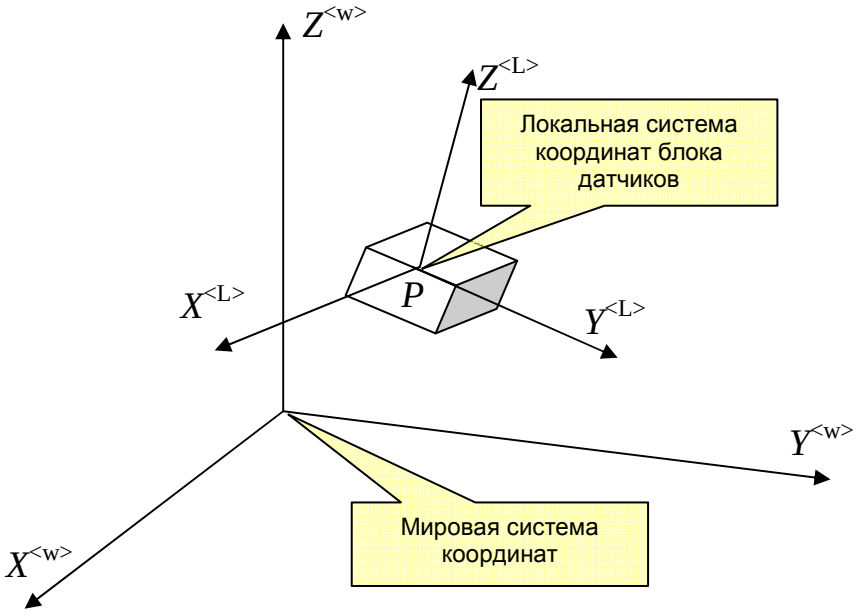

Рис. 67 Мировая система координат $X^{<w>}$, $Y^{<w>}$, $Z^{<w>}$ и система координат блока инерциальных датчиков $X^{<L>}$, $Y^{<L>}$, $Z^{<L>}$

Матрица поворота или матрица ориентации *M* имеет размер 3x3 и представляет собой набор из трех векторов ориентации осей $X^{<L>}$, $Y^{<L>}$, $Z^{<L>}$ локальной системы координат относительно осей мировой системы координат:

$$M = \begin{pmatrix} X_x^{<L>} & X_y^{<L>} & X_z^{<L>} \\ Y_x^{<L>} & Y_y^{<L>} & Y_z^{<L>} \\ Z_x^{<L>} & Z_y^{<L>} & Z_z^{<L>} \end{pmatrix}$$

В начальный момент времени матрица *M* равна единичной матрице:

$$M_0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Вектор положения $P=(P_x, P_y, P_z)$ определяет точку положения локальной системы координат относительно начала координат мировой системы координат.

Для перевода вектора направления $A^{<L>} = \left(A_x^{<L>}, A_y^{<L>}, A_z^{<L>}\right)$, заданного в локальной системе координат, в вектор направления в мировой

системе координат $A^{<w>} = \left(A_x^{<w>}, A_y^{<w>}, A_z^{<w>}\right)$, следует вектор-строку $A^{<L>}$ умножить на матрицу преобразования *M*:

$$\vec{A}^{<w>} = \vec{A}^{<L>} \times M$$

Для поворота матрицы *M* следует ее умножить слева на матрицу соответствующего поворота:

$$M := M_{\text{пов.}} \times M$$

Где: $M_{\text{пов.}}$ – матрица, соответствующего поворота (см. далее).

При реализации на языке программирования умножение матриц важно производить через промежуточную матрицу, т.к. в противном случае, если результат умножения сразу записывать обратно в матрицу *M,* программист неизбежно повреждает элементы матрицы *M* до того, как они были полностью использованы в операции умножения матриц.

Матрица поворота $M_{\text{пов.}}$ на угол α по часовой стрелке относительно оси *X*, будет называться $M_{OX}(\alpha)$, и имеет вид:

$$M_{\text{пов.}} = M_{OX}(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}$$

Матрица поворота $M_{\text{пов.}}$ на угол α по часовой стрелке относительно оси *Y*, будет называться $M_{OY}(\alpha)$, имеет вид:

$$M_{\text{пов.}} = M_{OY}(\alpha) = \begin{pmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{pmatrix}$$

Матрица поворота $M_{\text{пов.}}$ на угол α по часовой стрелке относительно оси *Z*, будет называться $M_{OZ}(\alpha)$, имеет вид:

$$M_{\text{пов.}} = M_{OZ}(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Во всех случаях синус и косинус угла рассчитывается заранее перед подстановкой в виде коэффициента в матрицу.

Направление осей вращения и перемещения

Перед применением устройства пользователь должен убедиться в положительных направлениях вращения и перемещениях по каждой оси. Следует учесть, что приводимые ниже формулы будут справедливы для тех устройств, чьи системы координат совпадают с системами координат

робота, а углы поворота вокруг осей откладываются по часовой стрелке. При наличии расхождений между данным правилом и реальным положительном направлением осей устройства, пользователь должен ставить в соответствующую величину знак минус.

Интегрирование

Трехосевой акселерометр инерциальной навигационной системы измеряет ускорение в локальной системе координат. А интегрирование ускорений и скоростей следует производить в мировой системе координат. Поэтому вектор «сырых» ускорений $\vec{a}^{<raw,L>} = \left(a_x^{<raw,L>}, a_y^{<raw,L>}, a_z^{<raw,L>}\right)$ из локальной системы координат следует перевести в мировую систему координат, учитывая калибровочные поправки $\vec{a}_0^{<raw>}$:

$$\vec{a}^{<raw,w>} = \left(\vec{a}^{<raw,L>} - \vec{a}_0^{<raw>}\right) \times M$$

Для интегрирования ускорений, следует сначала произвести перевод из «сырых» ускорений в ускорения, измеряемые в метрах в секунду. Для этого следует умножить на соответствующий коэффициент пропорциональности K_a. А также умножить на период расчета Δt. Важно также учесть нулевое значение ускорения свободного падения $\vec{g}_0^{<raw>}$:

$$\vec{v} := \vec{v} + K_a \cdot \Delta t \cdot \left(\vec{a}^{<raw,w>} - g_0^{<raw>}\right)$$

Где: $\vec{v}$ – вектор скорости в мировой системе координат. Изначально значение $\vec{v}$ должно равняться нулю.

Важно в роботе иметь механизм, позволяющий сбрасывать вектор скорости в ноль в те моменты, когда робот однозначно остановился. В противном случае накопленная ошибка не будет давать возможности адекватно ее проинтегрировать для получения положения. Формировать сигнал остановки и сброса можно также путем определения длительного отсутствия вибраций по показаниям акселерометра.

Также, если система имеет дополнительные средства измерения скорости, то скорость, измеряемую данным способом, следует периодически приравнивать к скорости по другим приборам.

Для получения координат робота в мировой системе координат относительно некоторой начальной точки, следует по следующей формуле:

$$\vec{P} := \vec{P} + \vec{v} \cdot \Delta t$$

Где: $\vec{P}$ – вектор положения робота в пространстве. Изначально равняется нулю или координатам начальной точки работы робота.

Несложно заметить, что двойное интегрирование может привести к большим погрешностям определения координат робота. Поэтому значение вектора $\vec{P}$ следует периодически приравнивать к известным координатам, определяемым каким-либо другим способом.

Трехосевой гироскоп инерциальной навигационной системы, позволяет определить угловые скорости вращения вокруг осей локальной системы координат датчика. А углы ориентации устройства формируются в мировой системе координат. Поэтому и интегрирование скоростей следует производить в мировой системе координат.

Для этого вектор «сырых» скоростей $\vec{\omega}^{<raw,L>}$, измеряемых гироскопами в локальной системе координат, следует перевести в мировую систему координат, используя текущую матрицу поворота M (полученную на предыдущем такте расчета). При этом важно учесть нулевое калибровочное значение вектора угловых скоростей $\vec{\omega}_0^{<raw>}$:

$$\vec{\omega}^{<raw,w>} = \left(\vec{\omega}^{<raw,L>} - \vec{\omega}_0^{<raw>}\right) \times M$$

Здесь: $\vec{\omega}^{<raw,w>}$ – вектор «сырых» скоростей в мировой системе координат.

После перевода вектора скоростей в мировую систему координат, можно произвести его интегрирование. Для этого сырые показания гироскопа умножаются на коэффициент пропорциональности K_ω, переводящего сырые показания датчиков в угловую скорость в радианах в секунду, а также на такт расчета Δt, измеряемый в секундах. В результате получается вектор приращения углов поворота $\Delta\vec{\alpha}$:

$$\Delta\vec{\alpha} = K_\omega \cdot \Delta t \cdot \vec{\omega}^{<raw,w>}$$

Полученные приращения углов следует применить к матрице поворота M:

$$M := M_{OZ}(\Delta\alpha_z) \times M_{OY}(\Delta\alpha_y) \times M_{OX}(\Delta\alpha_x) \times M$$

Несложно заметить, что в результате интегрирования может накапливаться ошибка, в результате чего матрица M не будет соответствовать реальной ориентации робота. В этом случае неправильно будет учитываться как направление действия линейного ускорения (а, следовательно, направления вектора скорости), а также неправильно будут рассчитаны оси вращения. Поэтому полученную матрицу ориентации следует периодически приводить к известной ориентации. В этом могут помогать как датчики магнитного курса, так и другие средства навигации.

4.2.10. Использование инерциальных навигационных систем в Dyn-Soft RobSim 5

В состав базы данных компонентов Dyn-Soft RobSim 5 входит несколько различных устройств на основе датчиков инерциальной системы навигации. Наиболее полный набор датчиков представлен в плате инерциальной системы навигации CHR-6dm AHRS. Данное устройство

содержит трехосевой гироскоп, трехосевой акселерометр и магнитометр. Кроме того, встроенный в устройство вычислитель позволяет вычислять ориентацию устройства в пространстве.

Для установки блока датчиков инерциальной системы навигации необходимо в 3D Studio MAX нажать кнопку «Сенсоры», находящуюся на панели RobSim5 (Рис. 68), и установить устройство на робота. В настройках компонента блока необходимо выбрать тип «Инерциальные системы навигации», а также выбрать конкретную модель устройства.

Устройство важно установить по центру робота (в противном случае датчик будет чувствителен к ускорениям, вызванными поворотом робота) и привязать к месту крепления (необязательно, если устройство размещается в главном корпусе робота).

Важна ориентация устройства. Желательно установить устройство так, чтобы ее оси совпадали с осями системы координат робота. В этом случае не возникнет проблем со знаком и направлением осей.

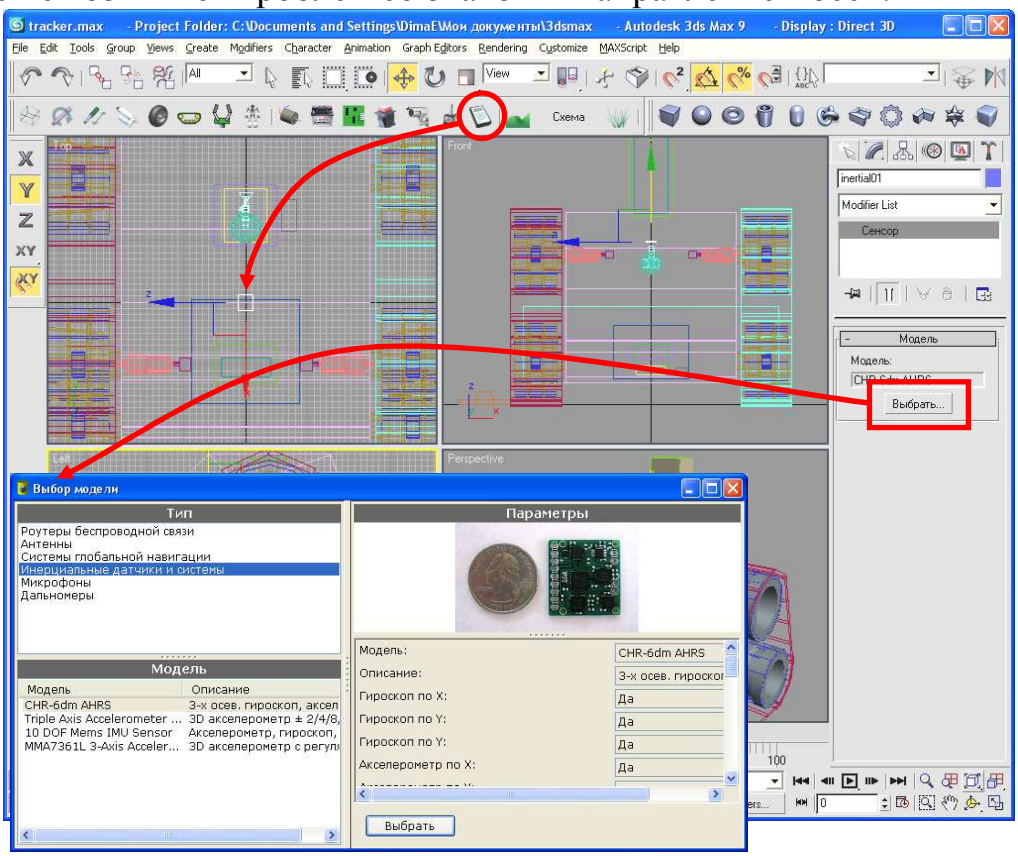

Рис. 68 Иллюстрация установки датчиков инерциальной навигационной системы в 3D Studio MAX

В зависимости от интерфейса связи с устройством, а также его способом питания, устройство следует подключить к бортовой ЭВМ или плате контроллера управления.

Блок датчиков инерциальной навигационной системы с интерфейсом UART можно напрямую подключить к UART-порту бортовой ЭВМ (при

его наличии) или микроконтроллеру AVR. Также можно предусмотреть на плате контроллера управления преобразователь интерфейсов UART-RS232 (микросхема MAX232 или ее аналог для уровня напряжений 3.3В – MAX3232), или UART-USB (микросхема FT232RL).

Если блок датчиков имеет интерфейс RS-232, то его удобно подключить напрямую к бортовой ЭВМ по COM-порту или использовать кабель-преобразователь интерфейсов USB-RS232, позволяющий подключить устройство к USB-порту бортовой ЭВМ.

В случае аналогового выхода, показания блока датчиков рекомендуется оцифровывать с помощью АЦП микроконтроллеров AVR. Из оцифрованного значения следует вычесть значение нулевого уровня и умножить на коэффициент пропорциональности между показанием датчика и показанием АЦП. При необходимости допускается производить интегрирование показаний датчиков на микроконтроллере AVR в условных единицах ускорения, а затем передавать на бортовую ЭВМ для домножения на коэффициент пропорциональности.

При наличии выхода типа «ШИМ», прием информации следует осуществлять в микроконтроллере AVR с помощью блока «Детектор ШИМ». Из измеренного значения интервалов логической единицы следует вычесть значение, соответствующее нулевого показанию датчика, а затем обработать сигнал по аналогии с аналоговым сигналом, описанным ранее.

Пример подключения блока датчиков платы CHR-6dm AHRS к UART-порту бортовой ЭВМ показан на Рис. 69

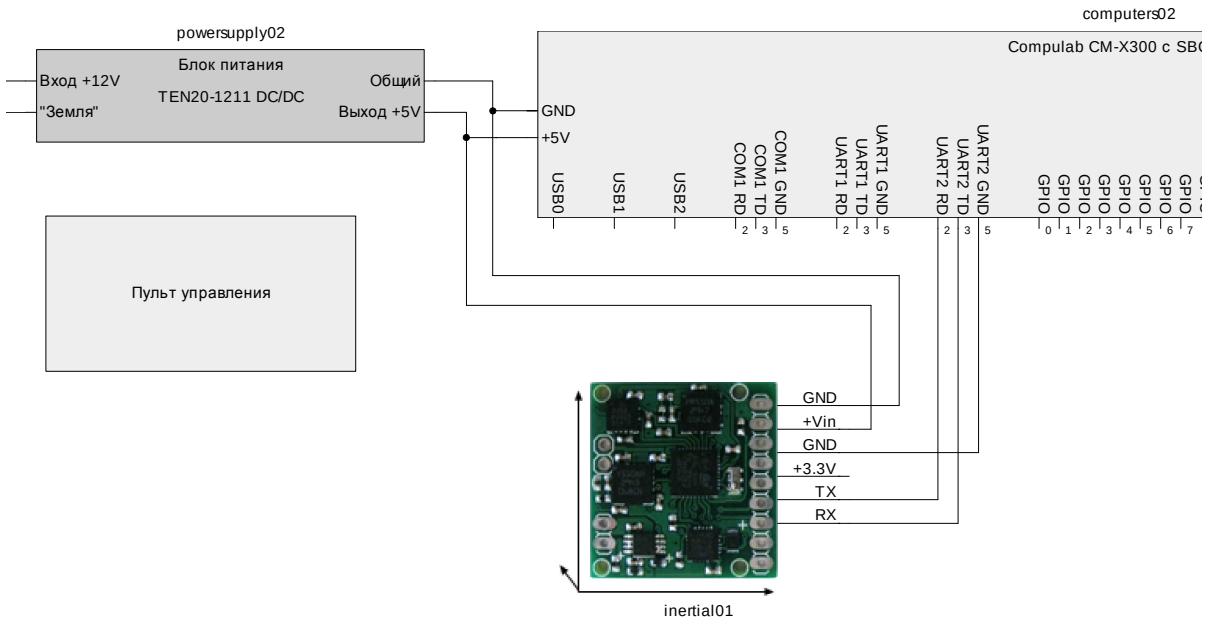

Рис. 69 Пример подключения датчиков инерциальной навигационной системы к бортовой ЭВМ в Dyn-Soft RobSim 5

Для обеспечения связи с платой датчиков инерциальной системы навигации следует настроить интерфейс связи на скорость 115200, 8 бит, 1

стоповый бит, без контроля четности, и настроить прием типа данных «Инерциальные данные» по каналу 0, а также передачу типа данных «Управл.инерциал.сист.» по любому каналу (Рис. 70).

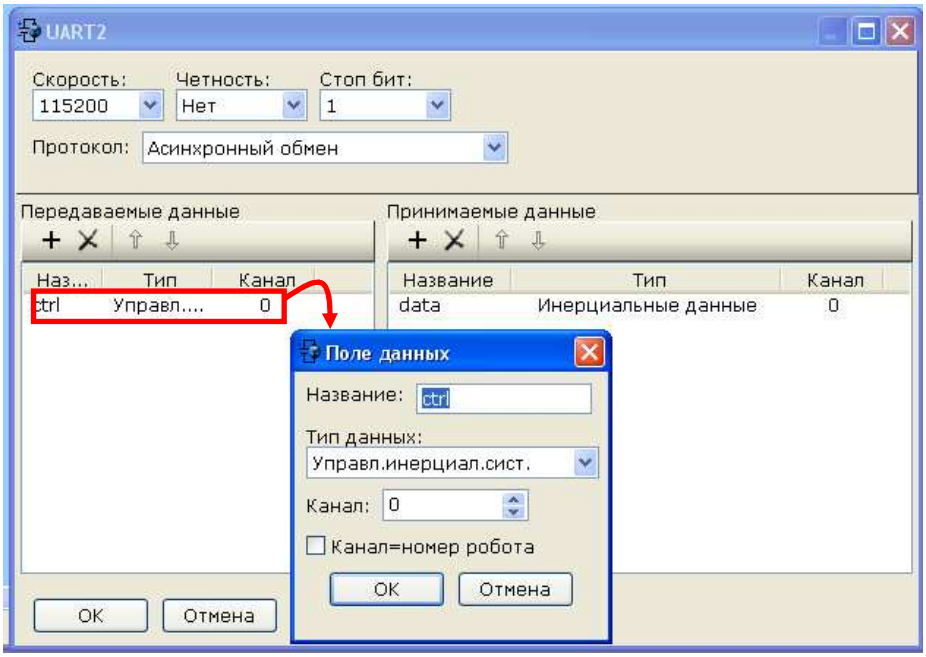

Рис. 70 Настройка UART-порта для связи с датчиком инерциальной системы в Dyn-Soft RobSim 5

Для кодирования и декодирования данных, принимаемых по интерфейсу связи с устройством, в структуру программного обеспечения необходимо установить блок «Драйвер инерциальной системы», который расположен на закладке «Преобразователи» редактора структурных схем программного обеспечения Dyn-Soft RobSim 5 (Рис. 71).

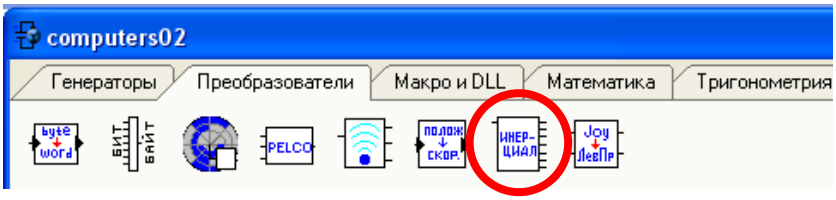

Рис. 71 Блок «Драйвер инерциальной системы» на палитре блоков редактора программного обеспечения бортовой ЭВМ

Порядок подключения данного блока показан на Рис. 72. Блок имеет вход данных, поступающих по интерфейсу связи от датчиков инерциальной системы (тип данных «Инерциальные данные»); выход управления блоком датчиков (тип данных «Управл.инерциал.сист.»), а также выходы, формирующие показания инерциальной системы.

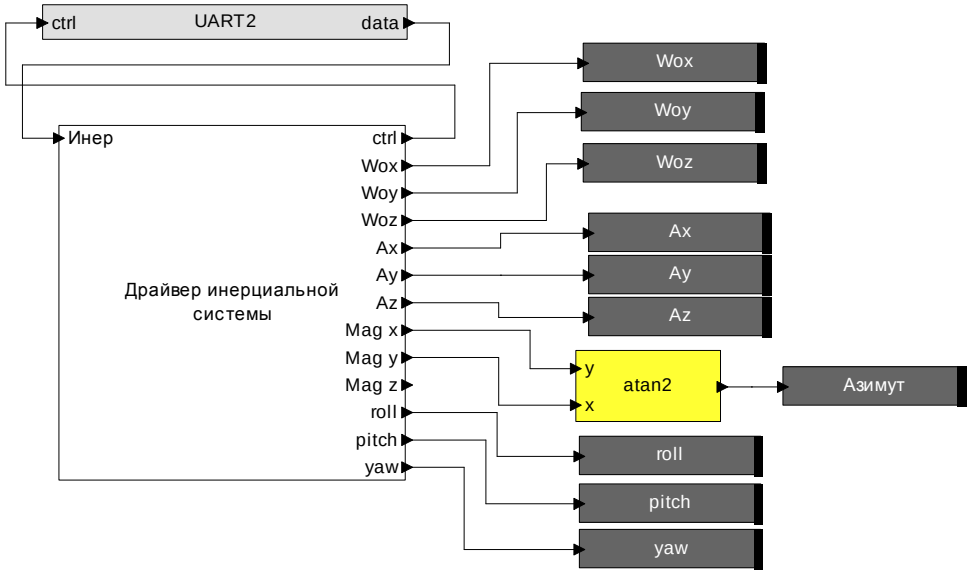

Рис. 72 Пример структуры программного обеспечения бортовой ЭВМ, обеспечивающий прием показаний датчиков инерциальной навигационной системы в Dyn-Soft RobSim 5

На выходе блока формируются следующие сигналы:

- Wox, Woy, Woz – (float) угловая скорость (в °/сек) вращения вокруг осей *X*, *Y* и *Z* соответственно. Данные возвращаются только при наличии гироскопа по соответствующим осям.
- Ax, Ay, Az – (float) ускорение (в mg) по осям *X*, *Y* и *Z* соответственно. Данные возвращаются только при наличии акселерометра по соответствующим осям.
- Mag x, Mag y, Mag z – (float) нормированное значение показания магнитного курса. В примере, с помощью блока atan2 по показаниям магнитного курса вычисляется направление на «сервер» (т.е. азимут со знаком «минус»). Данные возвращаются только при наличии магнитометра.
- roll, pitch, yaw – (float) вычисленные с помощью вычислителя блока датчиков углы (в градусах) ориентации устройства относительно на начального положения устройства, соответственно угол крена (вращение вокруг оси X), угол тангажа (вращение вокруг Y) и угол рыскания (вращение вокруг Z). Данные возвращаются только при наличии у устройства возможности вычисления данных величин.

Показания инерциальной систему навигации можно интегрировать или использовать для стабилизации каких-либо движений мобильного робота. Особенно удобно показание магнитного курса, с помощью которого можно достаточно легко вычислять азимут ориентации робота.

Процесс проведения экспериментальных исследований с применением инерциальной системы навигации на основе платы CHR-6dm

AHRS показан на Рис. 73. Важным критерием работы системы инерциальной навигации является отсутствие абсолютных нулей на выходе блока «Драйвера инерциальной системы».

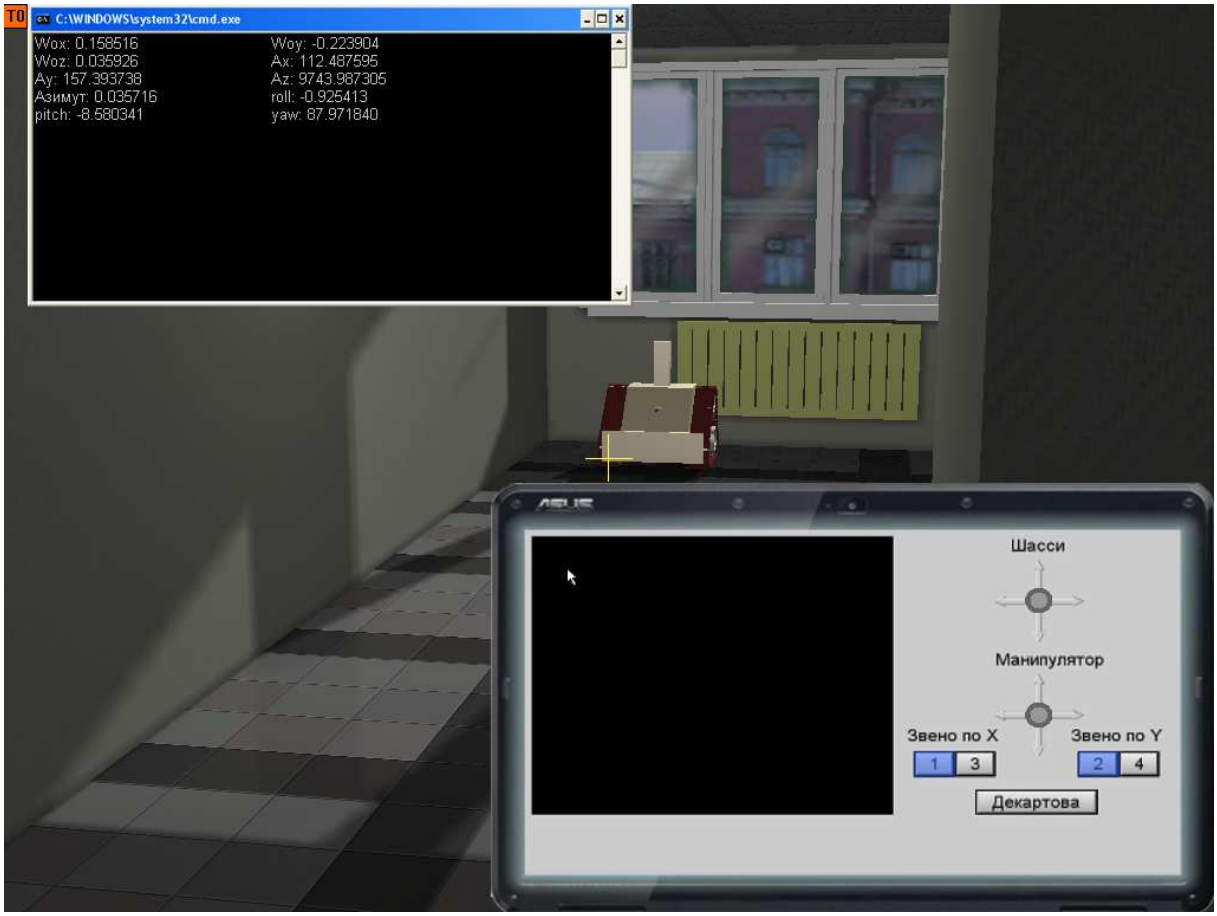

Рис. 73 Иллюстрация проведения экспериментальных исследований работы датчиков инерциальной навигационной системы

4.3.Навигация по датчикам обратной связи на колесах

4.3.1. Введение

На колесах робота обычно установлены датчики обратной связи, на основе которых осуществляется обратная связь для регуляторов скорости. Путем подсчета количества этих меток можно определить пройденный путь и углы ориентации робота.

Однако в силу эффектов проскальзывания и прокручивания колес точность навигации на основе этого метода не слишком велика. Но на малых расстояниях (до 1 метра), абсолютная погрешность данного метода приемлема.

Данный способ навигации удобно использовать совместно с другими способами навигации, например, для организации локальной карты местности, визуальной навигации, а также для отсчета прохождения небольших отрезков пути.

4.3.2. Принцип организации навигации по датчикам обратной связи для гусеничных и колесных машин, использующих для поворота разность скоростей левого и правого борта

Принцип организации навигации по датчикам обратной связи для гусеничных и колесных машин, использующих для поворота разность скоростей левого и правого борта достаточно прост. Для начала следует определить коэффициент пересчета K количества меток n в пройденный колесами робота путь L:

$$L = K \cdot n$$

Для колесных машин несложно увидеть, что при прохождении колесом одного оборота робот передвигается на величину $2\pi R$ (где, R – радиус колеса). Вал энкодера при этом поворачивается на i оборотов (где, i – коэффициент редукции от оси колеса до вала энкодера, если вал энкодера совмещен с осью колеса, то $i = 1$). Если на энкодере N меток на оборот, а энкодер двухканальный, то от каждой метки регистрируется 4 фронта, поэтому при одном обороте колеса энкодер регистрирует $4 \cdot N \cdot i$ метки. Поэтому коэффициент K для колесного робота определяется по формуле:

$$K = \frac{2 \cdot \pi \cdot R}{4 \cdot N \cdot i}$$

Для гусеничных машин формула для коэффициента K похожа, но учитывает толщину гусеницы и соотношение радиусов ведущего и опорного колеса:

$$K = \frac{2 \cdot \pi \cdot \left(R_{оп} + h_{гус}\right)}{4 \cdot N \cdot i} \cdot \frac{R_{оп}}{R_{вед}}$$

Где: $R_{оп}$ – радиус опорного колеса; $h_{гус}$ – толщина гусеницы; $R_{вед}$ – радиус ведущего колеса.

Радиусы колес, как для колесного, так и для гусеничного робота, удобно брать в тех единицах, в которых будет осуществляться дальнейшая навигация. Рекомендуется использовать сантиметры.

Суммировать метки следует не вдоль одной прямой, а в направлении движения робота. Для этого следует учитывать приращение числа меток за цикл расчета (циклом расчета может считаться любой интервал времени,

или любое периодически повторяемое событие, например, прием меток по COM-порту в бортовой ЭВМ).

Таким образом, следует найти приращение Δn_L, Δn_R в метках левого и правого борта:

$$\Delta n_L = n_L - n_{L,old}$$

$$\Delta n_R = n_R - n_{R,old}$$

$$n_{L,old} := n_L$$

$$n_{R,old} := n_R$$

Где: n_L, n_R – количество меток, зарегистрированных энкодером колес левого и правого борта соответственно; $n_{L,old}$, $n_{R,old}$ – число меток зарегистрированных на предыдущей итерации цикла (изначально они равны нулю). Под оператором «:=» подразумевается присвоение значения (чтобы выражение не выглядело равенством).

Путь ΔL, проходимый роботом за такт расчета, будет определяться, как среднее арифметическое значение пути, пройденным левым и правым бортом:

$$\Delta L = \frac{\Delta n_L + \Delta n_R}{2} \cdot K$$

В мировой системе координат путь ΔL откладывается вдоль направления движения робота (Рис. 74).

Приращения перемещения Δx и Δy соответственно по осям X и Y определяются по формулам:

$$\Delta x = \Delta L \cdot \sin\alpha$$

$$\Delta y = \Delta L \cdot \cos\alpha$$

Где: α – угол азимута ориентации робота (в радианах).

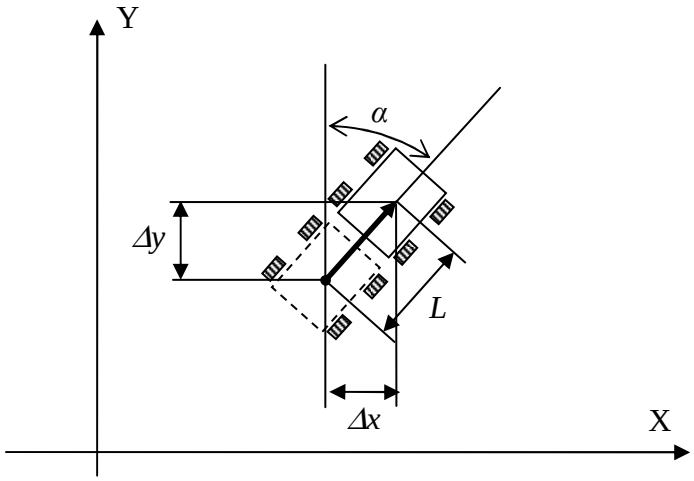

Рис. 74 Схема перемещения робота в мировой системе координат

Координата *x*, *y* робота определяются по формуле:

$$x := x + \Delta x$$

$$y := y + \Delta y$$

Также по приращению перемещения колес можно найти приращение угла поворота робота (Рис. 75).

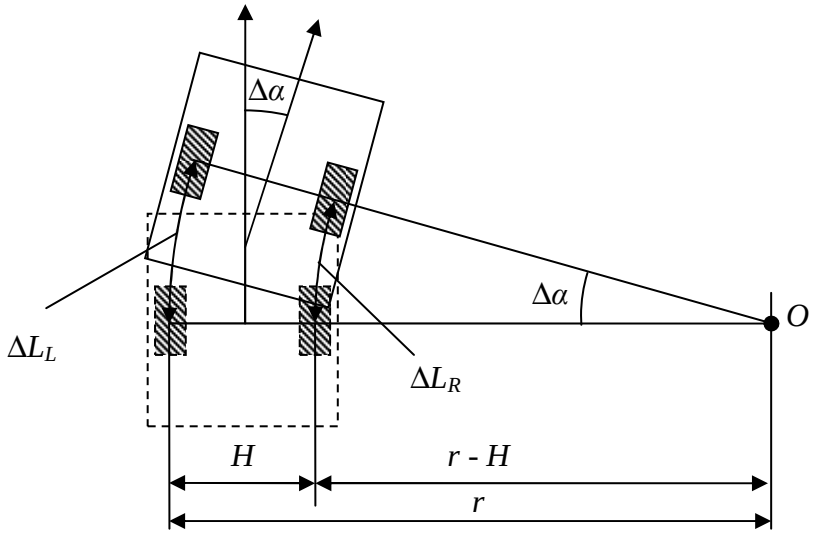

Рис. 75 Схема расчета приращения угла поворота робота

Для нахождения приращения угла поворота робота $\Delta\alpha$ (в радианах) определяется мгновенный центр вращения (точка *O*). Несложно увидеть равенство углов, обозначенных на рисунке $\Delta\alpha$. Также из рисунка видно, что путь (ΔL_L), проходимый левым колесом, можно определить как длину дуги, поэтому он равен:

$$\Delta L_L = r \cdot \Delta\alpha$$

Аналогичным образом путь (ΔL_R), проходимый правым колесом, равен:

$$\Delta L_R = (r - H) \cdot \Delta\alpha$$

Где: *H* – эффективная колесная база (поперечное расстояние между колесами).

Из этих двух равенств можно составить систему уравнений и найти $\Delta\alpha$:

$$\begin{cases} \Delta L_L = r \cdot \Delta\alpha \\ \Delta L_R = (r - H) \cdot \Delta\alpha \end{cases}$$

$$r = \frac{\Delta L_L}{\Delta\alpha}$$

$$\Delta L_R = \left(\frac{\Delta L_L}{\Delta\alpha} - H \right) \cdot \Delta\alpha = \Delta L_L - H \cdot \Delta\alpha$$

$$\Delta\alpha = \frac{\Delta L_L - \Delta L_R}{H}$$

Подставив вместо ΔL_L и ΔL_R приращения в метках, умноженных на коэффициент K, получается:

$$\Delta\alpha = (\Delta n_L - \Delta n_R) \cdot \frac{K}{H}$$

Таким образом, значение угла ориентации робота определяется выражением:

$$\alpha := \alpha + \Delta\alpha$$

Таким образом, получается следующая расчетная схема для навигации мобильного робота:

$$\Delta L = \frac{\Delta n_L + \Delta n_R}{2} \cdot K$$

$$\Delta\alpha = (\Delta n_L - \Delta n_R) \cdot \frac{K}{H}$$

$$x := x + \Delta L \cdot \sin\alpha$$

$$y := y + \Delta L \cdot \cos\alpha$$

$$\alpha := \alpha + \Delta\alpha$$

Несложно заметить, что приращения можно заменить дифференцированием, а увеличение x, y и α на Δx, Δy и $\Delta\alpha$ соответственно – интегрированием.

Ведь для дискретного случая дифференцирование это:

$$y = \frac{x - x_{old}}{\Delta t}$$

$$x_{old} := x$$

Где: x – вход дифференциального звена; y – выход дифференциального звена; Δt – такт расчета.

Операция интегрирования это:

$$y := y + x \cdot \Delta t$$

Где: x – вход интегрального звена, y – выход интегрального звена.

Используя эти свойства операций интегрирования и дифференцирования расчетную схему навигации можно записать через скорости:

$$v = \frac{\frac{\Delta n_L}{\Delta t} + \frac{\Delta n_R}{\Delta t}}{2} \cdot K$$

$$\omega = \left(\frac{\Delta n_L}{\Delta t} - \frac{\Delta n_R}{\Delta t} \right) \cdot \frac{K}{H}$$

$$x := x + v \cdot \Delta t \cdot \sin \alpha$$

$$y := y + v \cdot \Delta t \cdot \cos \alpha$$

$$\alpha := \alpha + \omega \cdot \Delta t$$

Или, используя строгую математическую запись:

$$v(t) = \frac{\frac{dn_L(t)}{dt} + \frac{dn_R(t)}{dt}}{2} \cdot K$$

$$\omega(t) = \left(\frac{dn_L(t)}{dt} - \frac{dn_R(t)}{dt} \right) \cdot \frac{K}{H}$$

$$\alpha(t) = \int_0^t \omega(\tau) d\tau$$

$$x(t) = \int_0^t v(\tau) \cdot \sin\left(\alpha(\tau)\right) d\tau$$

$$y(t) = \int_0^t v(\tau) \cdot \cos\left(\alpha(\tau)\right) d\tau$$

Здесь: v – мгновенная линейная скорость робота, всегда направленная вперед по направлению движения робота; ω – скорость поворота робота относительно его вертикальной оси (рад/сек).

4.3.3. Реализация навигации по датчикам обратной связи на колесах в среде Dyn-Soft RobSim 5

В Dyn-Soft RobSim 5 для реализации навигации по датчикам обратной связи на колесах необходимо в 3D Studio MAX использовать

двигатели со встроенным энкодером (Рис. 76), либо установить внешние инкрементные энкодеры на колеса. В последнем случае датчики нужно привязать к колесам путем выбора этих колес в параметрах датчиков в качестве измеряемой оси (Рис. 77).

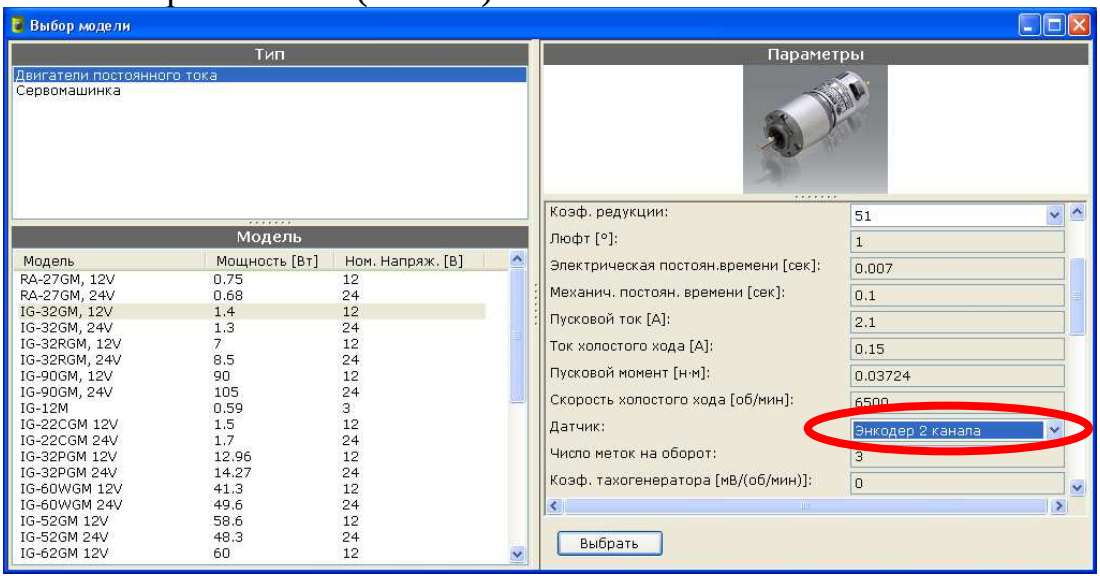

Рис. 76 Иллюстрация выбора двигателей со встроенным энкодером

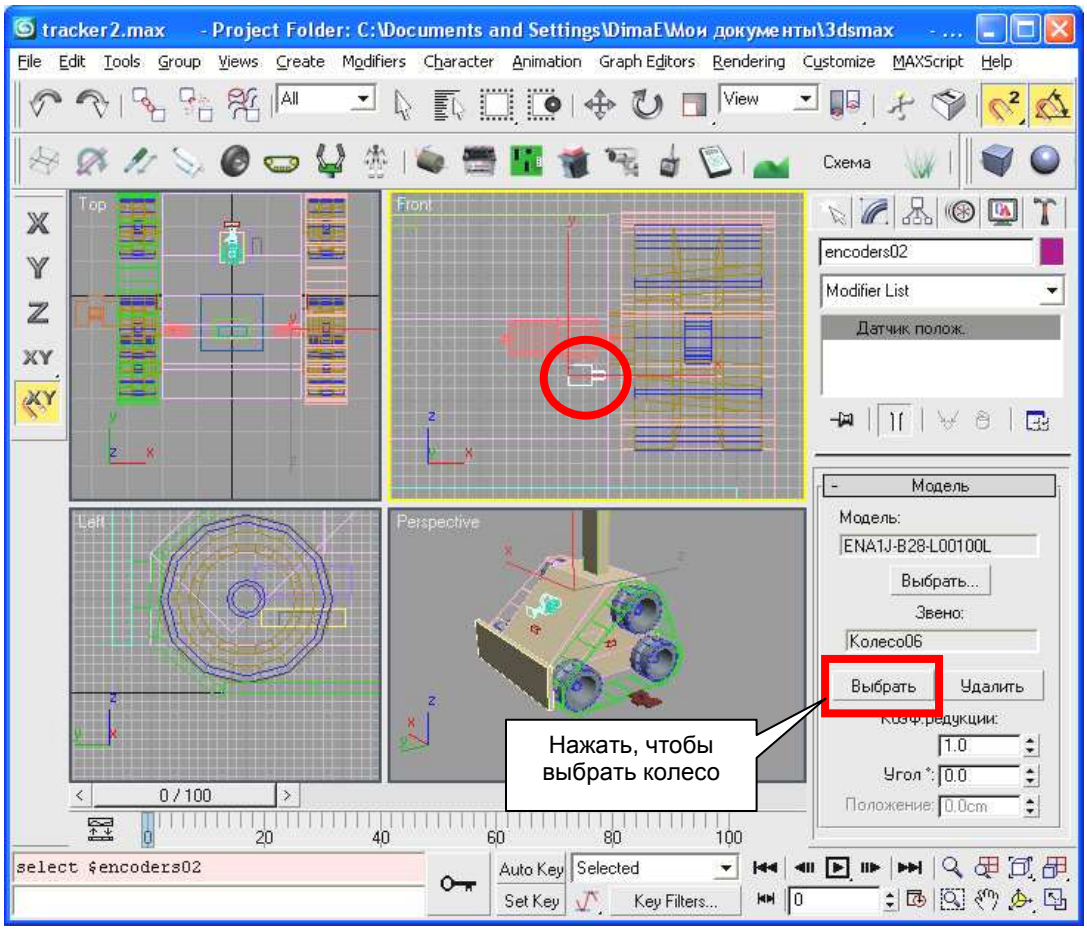

Рис. 77 Иллюстрация установки внешних энкоденов на колеса в 3D Studio MAX

В редакторе схем и подключений (кнопка «Схема») энкодеры следует подключить к источнику питания и к микропроцессору контроллера управления (Рис. 78).

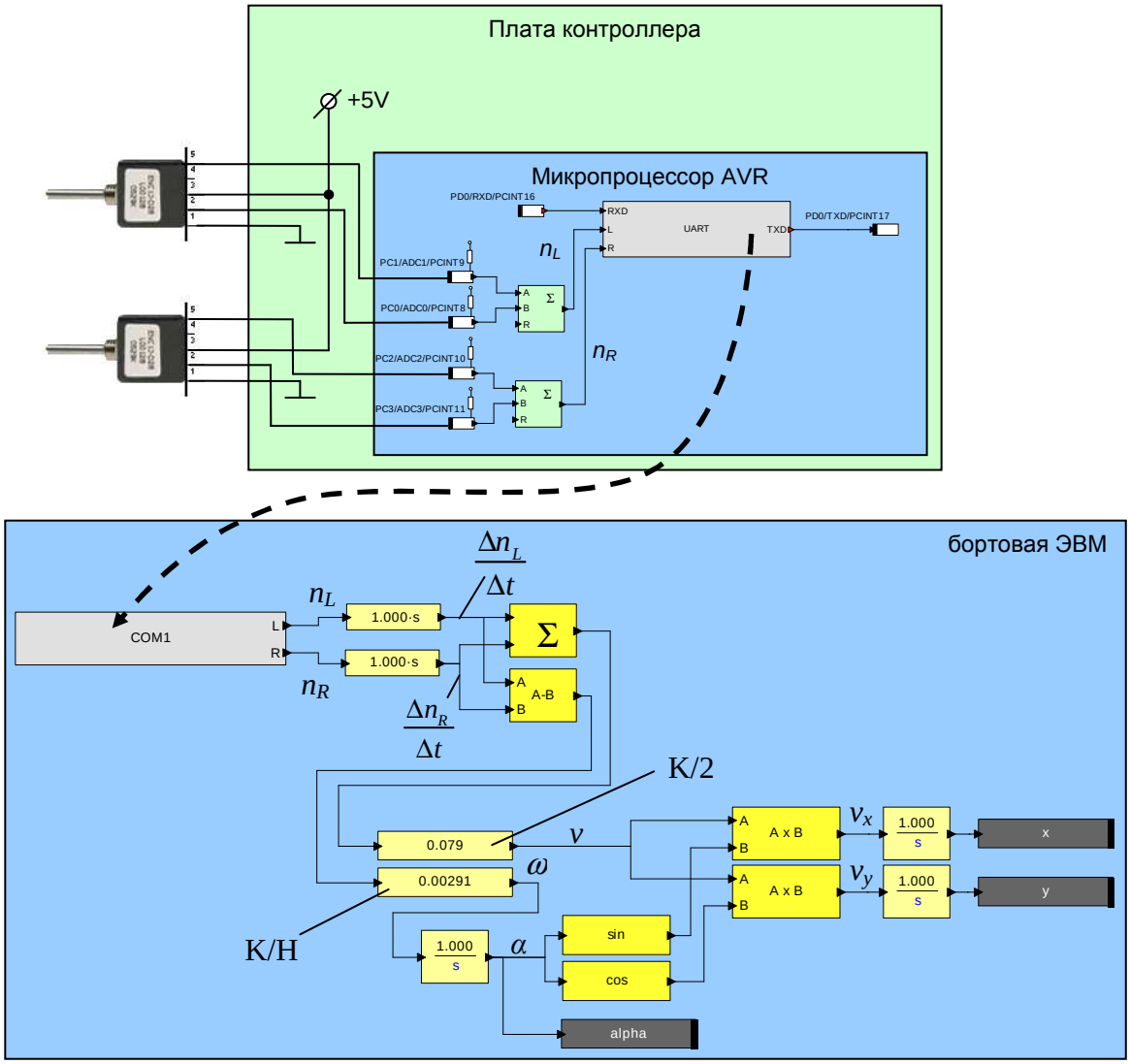

Рис. 78 Схема реализация навигации по датчикам на колесах в Dyn-Soft RobSim 5

При настройке программного обеспечения следует перевести ножки микропроцессора, на которые был подан сигнал с энкодеров, в режим «Вход с подтягивающим резистором». Используя блок «Счетчик для инкрементного датчика» можно произвести подсчет меток левого (n_L) и правого (n_R) энкодера (блок считает все 4 фронта метки, поэтому сумма будет в 4 раза больше числа меток инкрементного датчика). Подсчет меток можно производить в типе данных short (настраивается в свойствах блока счетчика). Подсчитанное число меток можно отправить на бортовую ЭВМ используя UART.

Обеспечив прием данных от микроконтроллера на бортовой ЭВМ, можно реализовать схему навигации, рассмотренную в главе 4.3.2. Для этого показания датчиков дифференцируются (рекомендуется

использовать реализацию дифференциатора в режиме «числа с плавающей запятой», задается в свойствах блока дифференцирования). Затем, согласно схеме в главе 4.3.2 сумму дифференциала меток следует умножить на $K/2$, чтобы получить скорость v, а разность на K/H, чтобы получить угловую скорость ω.

Интеграл от угловой скорости ω дает угол поворота робота α (в радианах). Получив синус и косинус от угла α, можно разложить скорость v на вектора v_x, v_y по осям мировой системы координат. Интеграл от v_x, v_y дает координаты x и y робота.

На Рис. 79 показан процесс испытания системы навигации по датчикам обратной связи на колесах. Для данного рода испытаний удобно использовать сцену «Office», в которой имеется пол, размер плиток которого 30x30 см. На такой сцене легко можно проверить адекватность навигации.

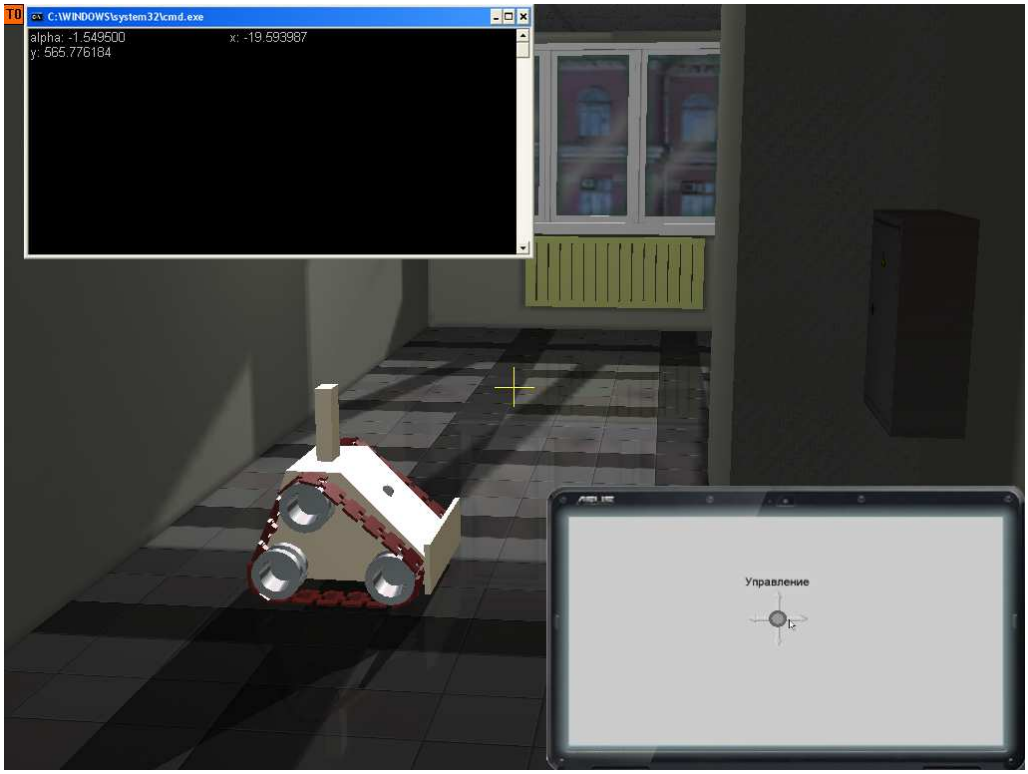

Рис. 79 Иллюстрация процесса испытания системы навигации по датчикам на колесах

4.4. Визуальная навигация

4.4.1. Введение

Под визуальной навигацией понимается навигация на основе видеокамеры, датчиков обратной связи на колесах, а также, опционально, дальномеров.

Визуальная навигация основана на сравнении текущего изображения с видеокамеры с ранее запомненными изображениями, сохраненными на карте местности. При таком сравнении производится коррекция угла ориентации робота, а также в некоторых случаях, координат *x*, *y* робота.

При наличии дальномеров также производится коррекция местоположения робота относительно статических стен, сохраненных на карте местности.

4.4.2. Режимы работы визуальной навигации

Алгоритм визуальной навигации имеет 4 режима работы:

- без навигации (режим «Н»);
- движение по карте (режим «Д»);
- запись карты (режим «З»);
- коррекция карты (режим «К»).

В режиме «без навигации» навигация робота полностью отключена.

В режиме «движение по карте» осуществляется навигация по ранее запомненной карте местности. В этом режиме робот осуществляет свою абсолютную привязку к карте местности по видеоизображению и показаниям дальномеров.

В режиме «запись карты» оператор робота проводит его обучение, в ручном режиме проводя его по маршрутам следования. В этом режиме навигация робота осуществляется исключительно по датчикам обратной связи на колесах. В процессе обучения визуально наблюдая за роботом и имея представления о его реальном положении и ориентации, оператор может производить коррекцию, как местоположения, так и угла ориентации робота на виртуальной карте местности, хранящейся в памяти робота.

Кроме того, в режиме записи карты игнорируются небольшие повороты в движении, т.к. считается, что этими поворотами оператор корректирует направление робота. Сложно предположить, что оператор робота в процессе обучения будет умышленно проводить робота по криволинейным маршрутам. Скорее всего, все повороты будут осуществляться на месте, а повороты в движении будут осуществляться только ради коррекции прямолинейного движения.

Пока робот двигается в режиме обучения, на карту в его памяти записываются отдельные кадры видеоизображения в текущие координаты робота (x, y, α). Запись кадра производится либо, если расстояние от предыдущего кадра становится больше заданной пороговой величины (обычно 25-30 см), либо степень похожести с ранее записанным изображением становится мало.

Помимо изображений, на карте местности отмечаются стены и статические препятствия, определяемые по дальномерам.

Также в процессе обучения оператор размещает на карте местности, так называемые, места. Впоследствии робот может получить команду прибыть в то или иное место. При этом узловые точки пути к данному месту будут совпадать с координатами изображений на карте. Это обеспечивает роботу движение по такой траектории, где будет обеспечена «узнаваемость» местности. Кроме того, если при обучении оператор смог провести робота по данной траектории, следовательно, такой путь должен существовать (временные препятствия в данный момент не рассматриваются).

В силу зависимости изображений местности от уровня освещенности, имеет смысл производить несколько серий обучения при различных освещениях. Так, например, днем окна на видеоизображении будут, скорее, светлыми, а вечером и ночью, скорее темными. Поэтому изображение местности кардинальным образом изменяется, и нужно делать серии обучения при этих условиях освещенности.

Режим «коррекция карты» работает аналогично режиму «запись карты» с тем отличием, что при обнаружении на карте старых похожих изображений и виртуальных стен их координаты обновляться. Режим коррекции необходим для исправления ошибок, допущенных оператором при обучении.

4.4.3. Принцип визуальной навигации

Принцип навигации предполагает, что следуя по маршруту, робот не может за короткое время достаточно сильно отклониться от ранее запомненной при обучении траектории. Поэтому, ориентируясь по датчикам обратной связи на колесах, робот приблизительно знает свое местоположение. В окрестности данной точки робот выбирает с карты местности из своей памяти несколько ранее запомненных изображений и сравнивает их с текущим изображением с видеокамеры при различном вертикальном и горизонтальном сдвиге (Рис. 80, а).

При этом определяется горизонтальный и вертикальный сдвиг Δu, Δv наиболее похожего изображения с карты относительно текущего видеоизображения.

Если камера обладает минимальным уровнем дисторсии объектива (минимальными искажениями), то сдвиг в пикселях прямо пропорционален углу поворота камеры. Зная горизонтальный сдвиг изображений Δu, и зная угол обзора камеры FOV (Рис. 80, б), легко определить угол дезориентации робота $\Delta\alpha$:

$$\Delta\alpha = \frac{\Delta u \cdot FOV}{W}$$

Здесь: W – ширина кадра видеоизображения в пикселях.

Величина $\Delta\alpha$ является углом отклонения от ранее запомненного изображения в данной точке. Поэтому коррекция угла поворота робота α осуществляется по формуле:

$$\alpha = \alpha_{\text{изб}} + \Delta\alpha$$

Где: $\alpha_{\text{изб}}$ – угол, хранящийся в координатах изображения на карте.

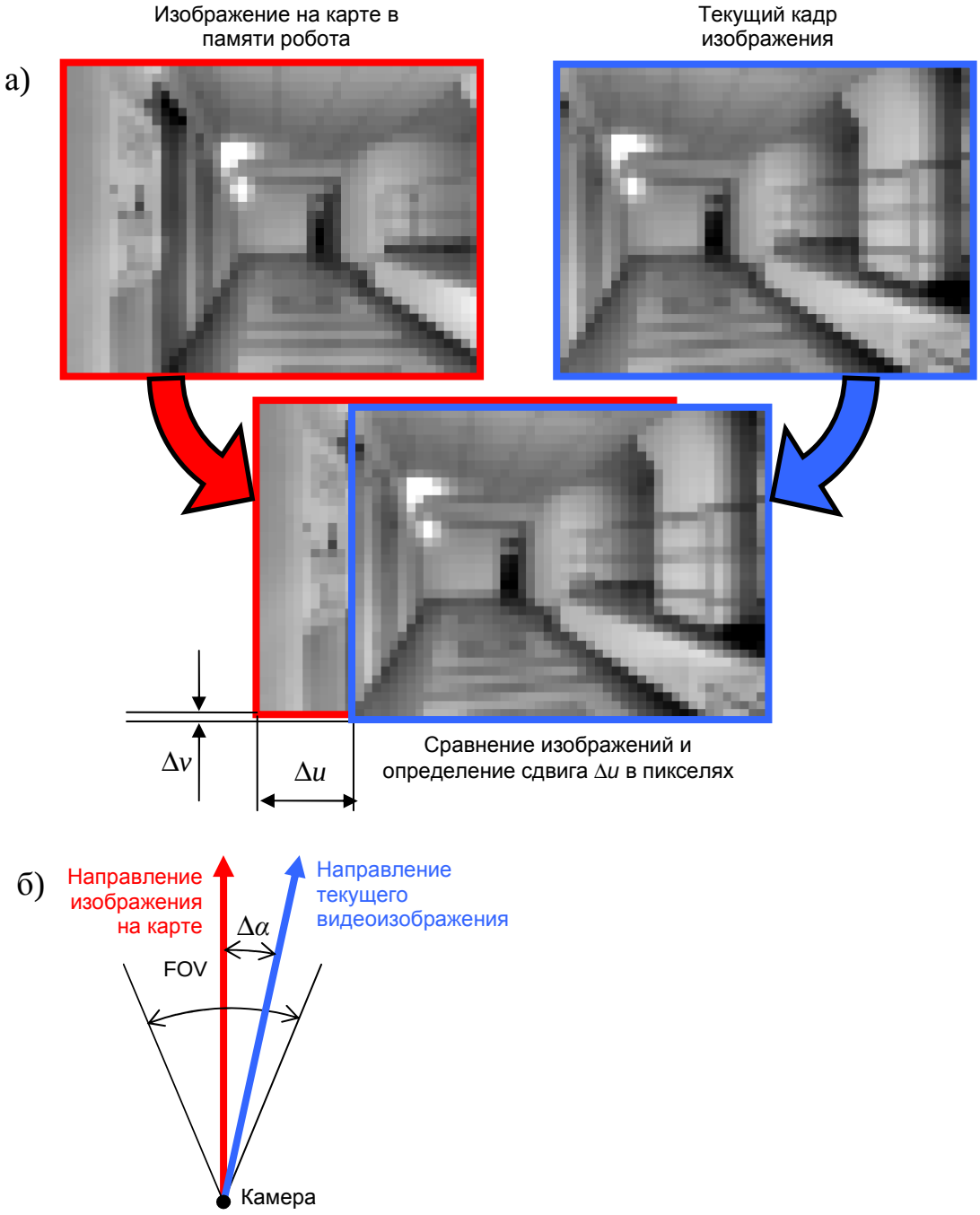

Рис. 80 Иллюстрация процесса визуальной навигации: а) процесс сравнения кадров и определения сдвига в пикселях; б) геометрическая модель пересчета сдвига в пикселях в приращение угла поворота

Размер кадра изображения на карте должен быть достаточно небольшим по нескольким причинам:

5. Во-первых, имеются ограничения на объем памяти, занимаемой картой.
6. Во-вторых, чем больше будет изображение, тем значимей на ней будут отдельные детали, в то время как роботу следует производить привязку к местности, а не к отдельным ее деталям.
7. В-третьих, обработка больших изображений требует больших вычислительных ресурсов, которых обычно у бортовой ЭВМ нет.

Рекомендуется использовать изображения всего 48 x 36 пикселей. Причем изображение перед записью на карту рекомендуется предварительно обработать: произвести коррекцию локальной яркости и контраста каждого пикселя. При сравнении изображение после такой обработки снижается влияние освещенности.

При наличии дальномеров можно составить карту расположения стен и статических препятствий. Путем сравнения карты расположения стен и текущего показания дальномеров можно производить коррекцию местоположения робота.

Так, например, на Рис. 81 изображена коррекция положения робота по правому дальномеру.

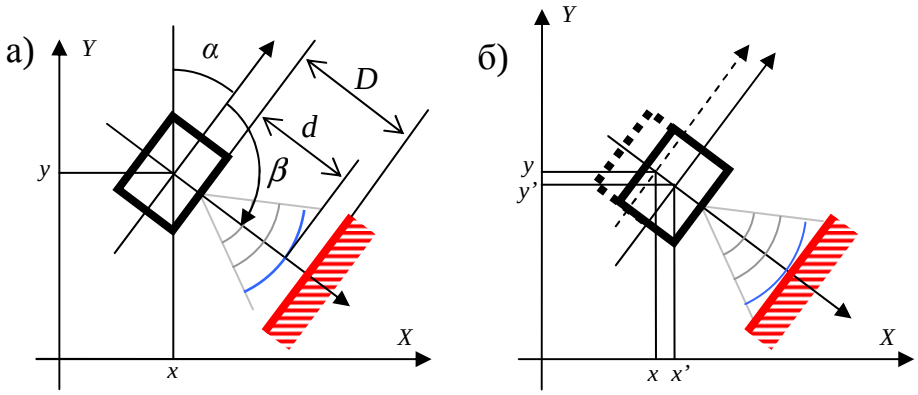

Рис. 81 Схема навигации по дальномерам: а) до коррекции; б) после коррекции

На рисунке показано, что в процессе движения робот находился в координатах x, y и двигался под углом α. При этом робот с помощью дальномера, находящегося под углом β по отношению к корпусу робота, обнаружил какое-то препятствие на расстоянии d. Сверившись с картой, робот обнаружил, что на его карте находится ранее запомненный элемент стены на расстоянии D, причем D ненамного отличается от d. Поэтому можно сделать вывод, что именно данный элемент стены был обнаружен дальномером, просто робот неправильно определяет свои координаты x, y.

Для коррекции координат робот использует следующие зависимости для определения своих новых координат *x’, y’*:

$$x' = x + (D - d) \cdot \sin(\alpha + \beta)$$

$$y' = y + (D - d) \cdot \cos(\alpha + \beta)$$

В силу того, что дальномером на роботе может быть несколько, а также препятствия могут быть случайно обнаруженными, то имеет смысл не мгновенно изменять координаты робота, а обеспечить постепенное приближение половинными шагами:

$$x' = x + (D - d)/2 \cdot \sin(\alpha + \beta)$$

$$y' = y + (D - d)/2 \cdot \cos(\alpha + \beta)$$

При наличии карты местности появляется возможность прокладывать пути к целевым точкам через места размещения изображений на карте местности. Для этого используется метод «бегущей волны» применительно к графу, составляемому из мест размещения изображений на карте. При этом при составлении графа вес ветви зависит не только от расстояния между местами размещения изображений на карте, но и от разности углов последовательных изображений. Это обеспечивает прокладывание пути без «срезания» углов.

На карте местности также могут отмечаться временные препятствия, определяемые каким-либо способом, например по дальномерам. Препятствия имеет смысл помечать на карте только при его обнаружении в непосредственной близости с роботом и при условии, что данное препятствие находится непосредственно перед роботом хотя бы несколько секунд.

У временных препятствий на карте местности имеется время жизни. Если наличие препятствия подтверждается датчиками робота, то время жизни препятствия обнуляется. В противном случае время жизни временных препятствий увеличивается до тех пор, пока оно не превысит пороговой величины, после чего препятствие исчезает с карты местности.

Временные препятствия блокируют возможность прокладывания пути к целевым точкам маршрута через места размещения этих временных препятствий. При появлении нового временного препятствия производится попытка заново проложить путь к целевой точке, с учетом наличия препятствий на пути. Если к целевой точке не удается проложить альтернативный маршрут, то формируется сигнал об ошибке.

4.4.4. Визуальная навигация в Dyn-Soft RobSim 5

В Dyn-Soft RobSim 5 для использования визуальной навигации имеется блок «Видеонавигация» в составе блоков редактора программного обеспечения бортовой ЭВМ на закладке «Интеллект» (Рис. 82).

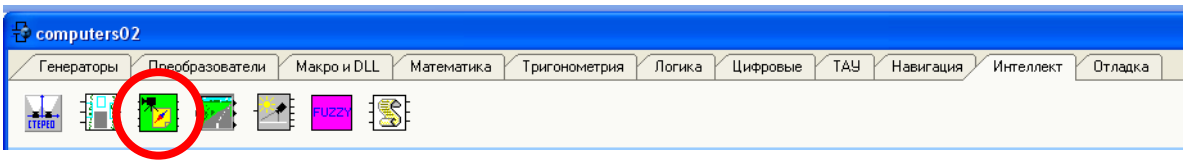

Рис. 82 Блок «Видеонавигация» на палитре блоков редактора программного обеспечения бортовой ЭВМ

Данный блок реализует все задачи визуальной навигации и формирует в своей памяти глобальную карту местности. Для хранения глобальной карты местности между сеансами запуска процесса моделирования, блок использует файл на локальном диске пользователя, задаваемый в настройке свойств блока.

На входе блока формирует сигнал «Глоб.карта» (глобальная карта местности), который совместим с сигналом «Видео», поэтому может отображаться на панели управления роботов в блоке «Видео».

Для использования блока «Видеонавигация» необходимо сначала реализовать схему навигации по датчикам обратной связи (см. главу 4.3.3), чтобы получить линейную скорость робота v (в см/сек) и угловую скорость ω (интегрировать их не требуется).

Блок «Видеонавигация» имеет следующие входы:

- *cam* – (видео) вход видеосигнала с камеры робота.
- *Vx*, *Vy* – (float) скорость движения (см/сек) робота в его локальной системе координат по оси X (направленной слева направо), и оси Y (направленной вперед). Сигналы формируются на основе системы навигации, на основе датчиков обратной связи на колесах.
- ω – (float) угловая скорость поворота робота (рад/сек, по часовой стрелке). Формируется также на основе датчиков обратной связи на колесах.
- *mode* – (char) режим работы системы визуальной навигации: 0 – режим без навигации («Н»); 1 – режим движения по карте («Д»); 2 – режим записи карты («З»); 3 – режим коррекции («К») (см. главу 4.4.2).
- *placeID* – (int) идентификатор целевой точки пути или 0. При формировании на данном входе значения, отличного от 0, блок «Видеонавигация» прокладывает путь вдоль набора запомненных изображений, и начинает формирование выходных сигналов *targetDist*, *pathA*, *pathX* (см. ниже). Идентификатор целевой точки пути должен формироваться все время, пока робот осуществляет движение. При сбросе сигнала в 0 путь сбрасывается.
- *addPlace* – (bit) по переднему фронту данного сигнала блок «Видеонавигация» записывает на карту новое место,

идентификатор которого возвращается на выходе *placeID*. Обычно, данный сигнал формируется с пульта управления роботом в режиме обучения.

- *man_dX, man_dY, man_dA* – сигналы коррекции местоположения робота на виртуальной карте местности, в памяти блока «Видеонавигация». Данные сигналы задают приращения местоположения робота по осям *X*, *Y* мировой системы координат и приращение к углу поворота соответственно. Сигналы формируются с пульта оператора и используются для коррекции местоположения робота во время его обучения. После коррекции угла поворота робота, автоматически осуществляется привязка к углам, кратным 45º. Т.е., если после коррекции угла угол становится примерно равен 90º, то он автоматически становится равным 90º. Это сделано умышленно, исходя из предположения о том, что в помещениях углы прямые, и пользователь вряд ли будет прокладывать маршруты не под прямыми углами.
- *clear* – (bit) сигнал сброса глобальной карты местности. После его формирования глобальная карта местности удаляется, координаты робота на ней становятся равными 0, 0, а ориентация робота становится равной 0 градусов (направление на «север»).
- $dist_{xx}$ – (float) входы сигналов дальномеров в сантиметрах. Разработчик в свойствах блока «Видеонавигация» самостоятельно определяет количество и название данных входов по числу дальномеров, используемого на роботе.
- *obs* – (bit) данный сигнал формирует на локальной карте местности отметку о временном препятствии прямо по курсу движения робота. Если при формировании сигнала *obs* в данной точке карты уже имеется препятствие, то новое препятствие не создается, а сбрасывается в 0 время жизни найденного препятствия. Разработчик должен формировать данный сигнал только в режиме движения по карте местности. Сигнал должен формироваться с учетом сигналов $isWall\_{xx}$, которые информируют о наличии стен на карте. Т.е., если дальномер определяет препятствие прямо по курсу движения робота, и на карте в этом месте нет стены, только в этом случае должен формироваться сигнал *obs*. После формирования на карте временного препятствия, блок «Видеонавигация» автоматически пытается проложить альтернативный маршрут к целевой точке (если она есть) с учетом данного препятствия пути.

На выходе блока «Видеонавигация» формируются следующие сигналы:

- *globalMap* – (тип данных «Глоб.карта») выход глобальной карты местности для передачи ее на пульт оператора.
- *robotLost* – (bit) сигнал, информирующий о том, что робот «заблудился». Сигнал формируется тогда, когда робот «не узнает» по видео свое местонахождения. Интеллектуальная система управления в данном случае должна предпринять действия по обнаружению местоположения робота. Робот должен, например, оглядеться вокруг или отъехать назад в то место, где сигнал *robotLost* еще не формировался. Следует отметить, что данный сигнал формируется не сразу, а спустя некоторый интервал времени после того, как робот не может себя обнаружить на карте местности. В принципе, используя навигацию по датчикам обратной связи, робот даже в режиме «robotLost» не может достаточно далеко отклониться от своего местоположения на виртуальной карте местности, поэтому некоторое время движение в режиме «robotLost» возможны.
- *robotX*, *robotY* – (float) координаты положения робота на глобальной карте местности по осям *X* и *Y* соответственно (см).
- *robotA* – (float) угол ориентации робота на глобальной карте местности (рад).
- *placeID* – (int) идентификатор последнего добавленного на карту места. Добавление отметки места производится путем формирования сигнала *addPlace*. После этого на выходе placeID формируется идентификатор этого места. Данный сигнал предназначен для формирования номера последнего добавленного места на пульте оператора.
- *targetDist* – (float) расстояние (см) до целевой точки или -1, если целевой точки нет. Сигнал формируется только при наличии на входе *placeID* сигнала с идентификатором целевого места назначения (не следует путать вход *placeID* и одноименный выход). Если к целевой точке невозможно проложить путь, то на выходе *targetDist* формируется -1. Также после достижения роботом целевой точки на данном выходе также формируется -1. Интеллектуальная система управления роботом должна начинать движение к целевой точке только при положительном значении данного сигнала.
- *pathA* – (float) пеленг текущего отрезка пути (рад). В непосредственной близости от целевой точки назначения (в радиусе ближней зоны, задаваемой в настройках блока) данный сигнал указывает целевой пеленг робота (т.е. пеленг, рассчитанный из ориентации целевого места назначения). В противном случае, сигнал *pathA* показывает рекомендуемое

направление поворота робота, обеспечивающее его движение вдоль проложенного пути.

– *pathX* – (float) линейное отклонение робота от текущего отрезка пути (см).
– *isWall_xx* – (bit) сигнал о наличии стен, найденных на карте по текущим показаниям дальномеров. Число данных выходов определяется числом дальномеров, заданных в свойствах блока. Данные сигналы удобно применять для формирования сигналов о наличии динамических препятствий на карте.

Окно настроек свойств блока «Видеонавигации» показано на Рис. 83.

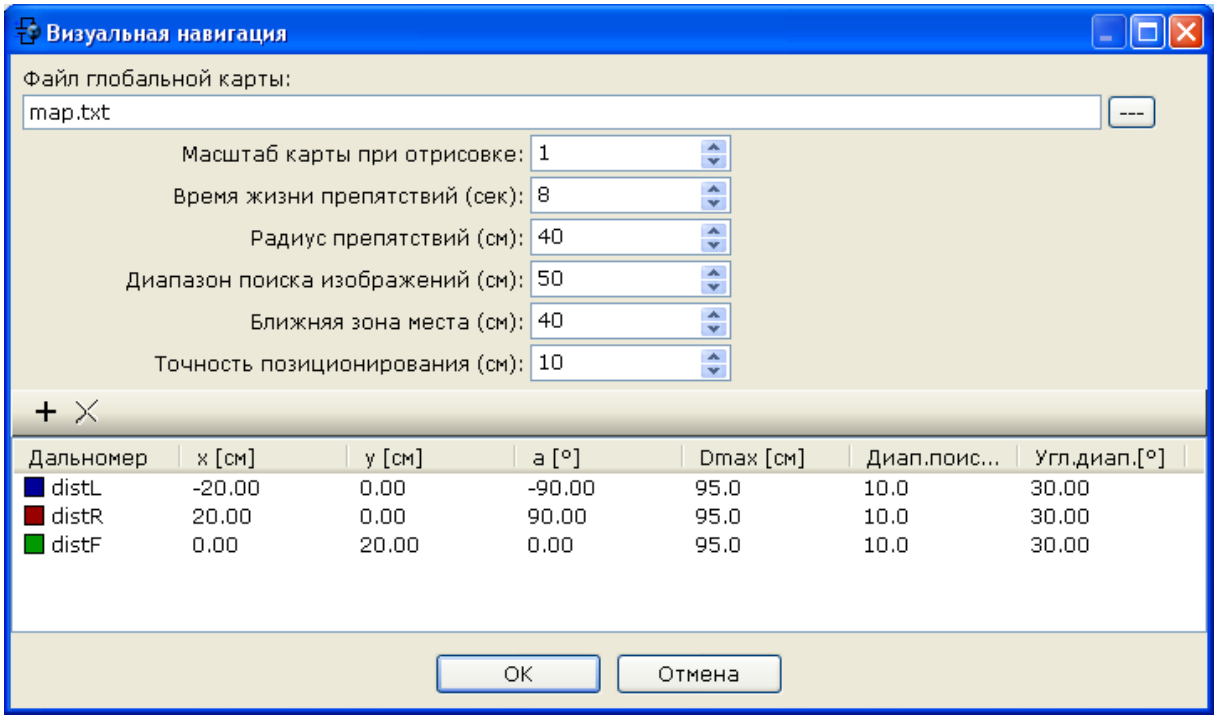

Рис. 83 Внешний вид окна редактора настроек блока «Видеонавигация»

В данном окне в поле «Файл глобальной карты» необходимо указать имя файла, в котором будет храниться глобальная карта местности между сеансами запуска Dyn-Soft RobSim 5. Если к файлу не задан путь, то файл хранится в папке Robots программного комплекса Dyn-Soft RobSim 5. Для обеспечения переносимости модели робота с одного компьютера на другой, рекомендуется оставлять данный файл в папке Robots.

Поле «Масштаб карты при отрисовке» задает относительный масштаб, в котором глобальная карта местности будет рисоваться в блоке «Видео» на пульте управления роботом. Данный масштаб зависит от субъективного представления пользователя о предполагаемых размерах карты и той ее части, которая будет отображаться на пульте управления. Если пользователь считает, то следует отображать карту в более мелком масштабе, он должен уменьшить данный коэффициент.

Поле «Время жизни препятствий» задает время существования временных препятствий на карте местности (в секундах). Как уже отмечалось, при формировании сигнала *obs* на глобальной карте местности формируется временное препятствие. Препятствие будет существовать на карте указанное в данном поле время, после чего оно пропадет с карты.

Поле «Радиус препятствия» определяет радиус создаваемых временных препятствий на глобальной карте местности.

Поле «Диапазон поиска изображений» задает радиус зоны вокруг робота на глобальной карте местности, из которой берутся изображения для организации процесса визуальной навигации. С одной стороны, чем больше будет данный радиус, тем больше будет шансов обнаружить изображение, похожее на текущее изображение с камеры, даже если предполагаемые координаты робота далеки от реальных. Но с другой стороны, чем больше данный радиус, тем больше вероятность ошибки (ведь, далекие изображения могут оказаться похожими друг на друга), и тем больше система будет тратить времени на сравнение, т.к. изображений в большом радиусе будет много.

Поле «Ближняя зона места» определяет радиус ближней зоны вокруг целевого места назначений. При движении по карте сигнал *pathA* в ближней зоне меняется с указания пеленга текущего отрезка пути, на пеленг целевой точки назначения.

Поле «Точность позиционирования» задает радиус зоны целевой точки. Если расстояние до целевой точки становится меньше указанного радиуса, и ориентация робота отличается от целевой ориентации менее 10º, то целевая точка считается достигнутой. В этом случае проложенный путь сбрасывается, а сигнал *targetDist* становится равным -1.

Ниже в окне имеется список дальномеров. Разработчик может добавлять, удалять или редактировать свойства каждого дальномера. При редактировании дальномера открывается окно, показанное на Рис. 84.

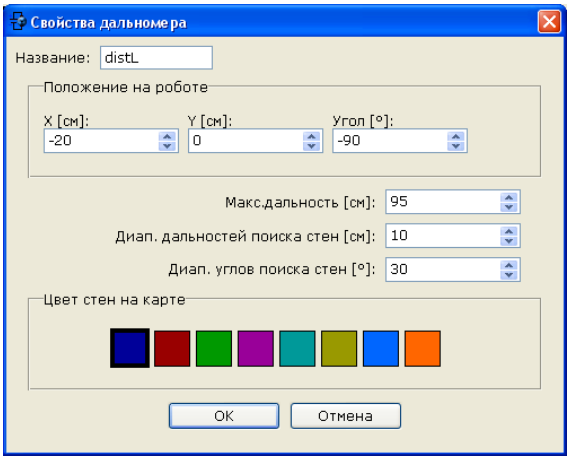

Рис. 84 Внешний вид окна редактирования свойств дальномера

По каждому дальномеру задается:

- Название, как оно будет отображаться у блока «Видеонавигация».
- Местоположение (в см) дальномера на роботе в его локальной системе координат (есть X – вправо, ось *Y* – вперед).
- Угол ориентации дальномера в локальной системе координат робота (по часовой стрелке, нулевой угол соответствует направлению вперед)
- Максимальная дальность (см), выше которой считается бесконечность (низко установленные дальномеры могут начать измерять расстояние до точки отражения от пола).
- Диапазон дальностей поиска стен. Данный параметр задает диапазон рассогласования между показанием дальномера и расстоянием до стены на глобальной карте местности, при котором считается, что именно данная стена сейчас измеряется дальномером. При таком обнаружении местоположения робота автоматически корректируется.
- Диапазон углов поиска стен (º). Параметр аналогичен предыдущему, только применяется для рассогласования по углам ориентации.
- Цвет для рисования стен, обнаруженных данным дальномером, при рисовании карты местности на пульте управления.

Пример подключения блока «Видеонавигация» показан на Рис. 85.

В данном примере от пульта управления по Ethernet передаются сигналы скорости движения левого и правого борта *L*, *R*. Данные сигналы передаются по COM-порту на плату контроллера управления.

Контроллер управления передает по COM-порту на бортовую ЭВМ показание инкрементных энкодеров на колесах робота (сигналы *L*, *R* с выхода COM-порта).

Бортовая ЭВМ реализует навигацию по датчикам обратной связи на колесах. Блоки, реализующие данный вид навигации, обведены пунктирной линией.

Также к роботу подключены дальномеры, размещенные спереди, с левого и правого борта робота. В данном примере использованы дальномеры с совмещенным каналом управления (см. главу 2.2.4). Показания дальномеров в «сыром» виде передаются по COM-порту от контроллера управления. Бортовая ЭВМ переводит «сырые» показания дальномеров в сантиметры, путем умножения на константу.

Сигнал с видеокамеры, подключенной к USB0, поступает на вход блока «Видеонавигация», а также передается на пульт управления роботом по Ethernet.

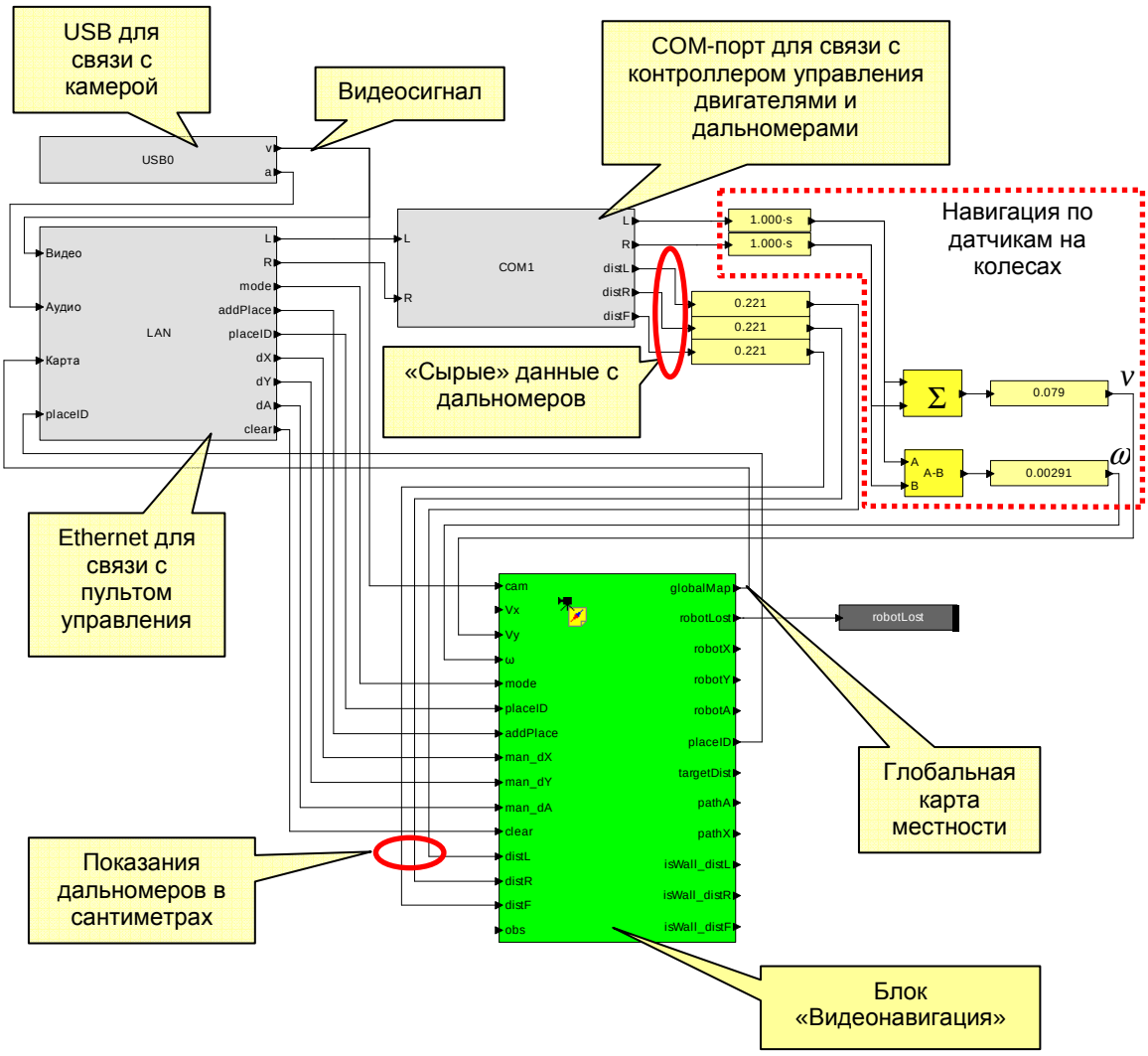

Рис. 85 Пример подключения блока «Видеонавигация» в структуру программного обеспечения бортовой ЭВМ мобильного робота в Dyn-Soft RobSim 5

С пульта управления также поступают сигналы управления режимом работы визуальной навигации (mode), идентификатор целевого места назначения (placeID), сигналы коррекции местоположения робота на карте dX, dY, dA, а также сигнал сброса глобальной карты местности.

Блок «Видеонавигация» формирует глобальную карту местности и отправляет ее по Ethernet на пульт управления.

Ответная структурная схема программного обеспечения пульта управления роботом показана на Рис. 86.

На данной схеме джойстик 1 предназначен для управления роботом в ручном режиме. Джойстик 2 и ползунок 1 предназначены для коррекции местоположения робота на карте. Выходы этих органов управления передаются по Ethernet на бортовую ЭВМ и подключены непосредственно к блоку «Видеонавигация» на входы *man_dX*, *man_dY* и *man_dA*.

Группа кнопок «НетНавигации», «Движение», «Запись» и «Коррекция» являются фиксируемыми кнопками и имеют общую группу.

Т.е. при нажатии одной из кнопок, она остается нажатой, а все остальные кнопки группы отжимаются. Таким образом, нажатой может быть только одна кнопка данной группы. С использованием приоритетного шифратора признаки нажатия на кнопку переводится в номер режима 0, 1, 2, 3. Номер режима передается через Ethernet на блок «Видеонавигация» на его вход *mode*.

На индикаторе «ВыборМеста» на панели управления отображается номер выбранного пользователем места назначения. Индикатор отображает сигнал с выхода блока счетчика. Счетчик управляется с помощью кнопок «Плюс» и «Минус». Данными кнопками пользователь может выставить значение счетчика. С помощью кнопки «Идти» текущее значение счетчика защелкивается в D-триггере, формируя тем самым идентификатор места назначения. Данный сигнал отправляется через Ethernet на блок «Видеонавигация» на вход *placeID*.

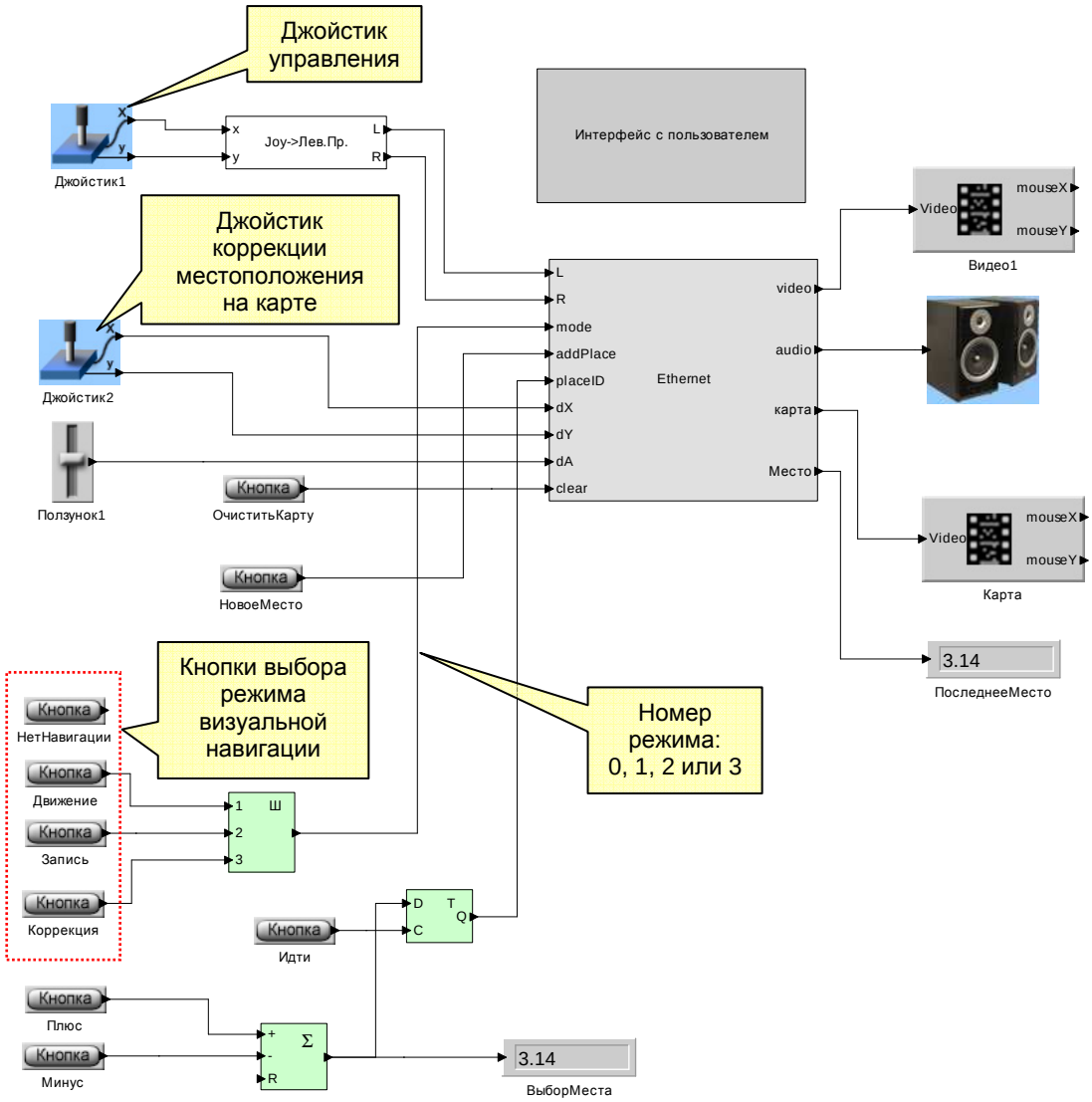

Рис. 86 Пример структуры программного обеспечения пульта управления для контроля визуальной навигации

Кнопка «ОчиститьКарту» формирует сигнал сброса глобальной карты местности. Признак нажатия на данную кнопку отправляется через Ethernet на блок «Видеонавигация» на его вход *clear*.

Кнопка «НовоеМесто» формирует сигнал создания нового места назначения на карте. Этот сигнал передается через Ethernet на блок «Видеонавигация» на вход *addPlace*.

По Ethernet от бортовой ЭВМ принимаются сигналы «Видео» и «Аудио» с камеры робота, глобальная карта местности (тип данных «Глоб.карта»), которая отображается на блоке «Видео», а также идентификатор последнего добавленного места назначения, который отображается на индикаторе пульта управления.

Внешний вид пульта управления с отображаемой на ней глобальной карты местности показан на Рис. 87.

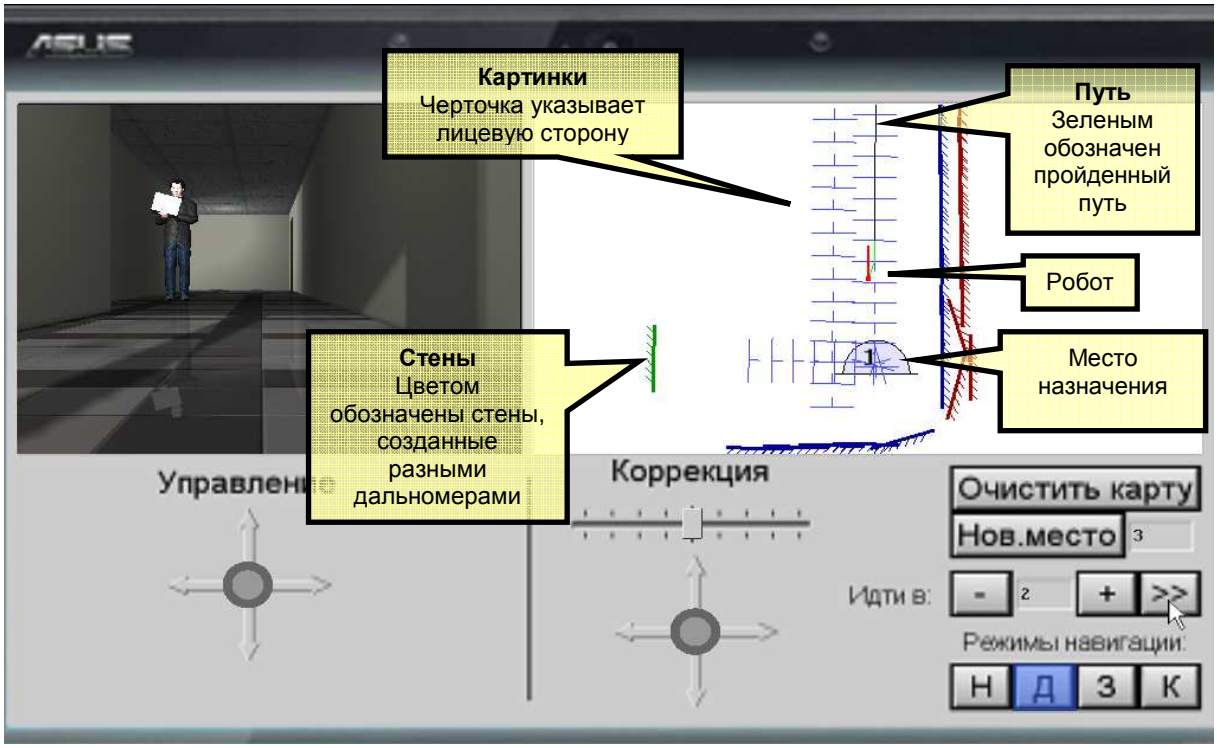

Рис. 87 Внешний вид пульта управления робота с отображением на нем глобальной карты местности

4.4.5. Проведение экспериментальных исследования визуальной навигации в Dyn-Soft RobSim 5

Процесс проведения экспериментальных исследований работы визуальной навигации показан на Рис. 88.

При проведении экспериментальных исследований разработчику следует сначала отладить навигацию по датчикам обратной связи на колесах. Признаком работы навигации является формирование адекватных углов поворота робота на глобальной карте. Рекомендуется проверить углы 90º и 180º.

Затем разработчик должен обучить робота. Для этого следует нажать кнопку «З» (запись) и провести робота по маршрутам следования в ту и другую сторону, не забывая пользоваться органами управления коррекции местоположения и угла поворота робота. В силу того, что запись карты производится только по датчикам обратной связи на колесах, качество навигации в этом режиме может оставлять лучшего. Поэтому пользователю следует периодически корректировать робота. Особенно после поворотов.

При записи карты пользователю следует отмечать на карте места назначения кнопкой «Нов.место».

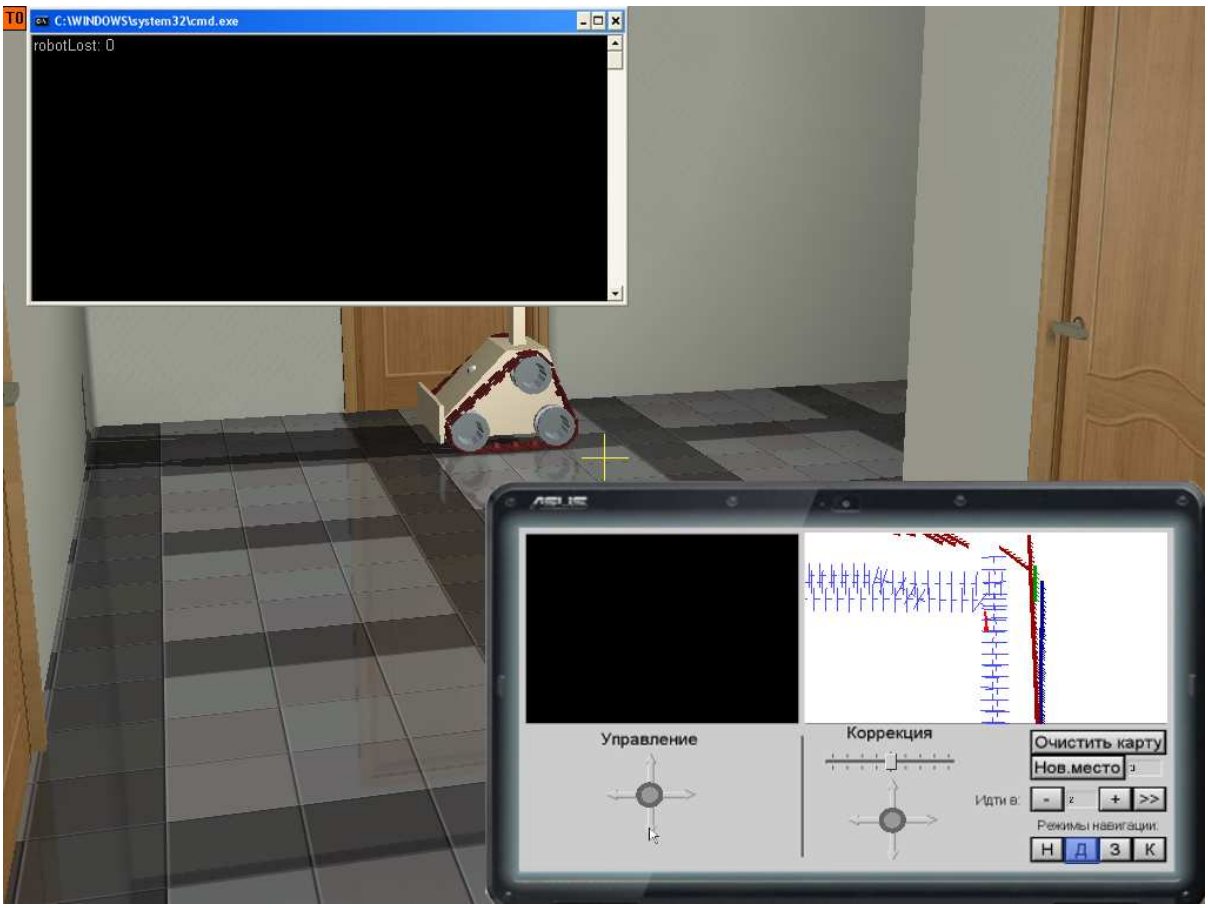

Рис. 88 Иллюстрация проведения экспериментальных исследований визуальной навигации

После записи карты, робота можно переключить в режим «Д» (движение по карте) и проверять работоспособность визуальной навигации.

Несложно заметить, что при работе в режиме «Д» производится коррекция местоположения и ориентации робота. Достаточно переместить робота в слегка неправильные координаты, то при подъезде к проложенным маршрутам робот корректируется.

5. Интеллектуальные системы управления

5.1. Технологии интеллектуальных систем управления

Система называется интеллектуальной, если она обладает следующими отличительными особенностями:

1. Система обладает знаниями, которые определяют ее поведение в сложной и заранее неизвестной ситуации.
2. Система имеет механизм логического вывода выходных сигналов на основе входных данных и знаний, заложенных в систему.
3. Система имеет механизм обучения или самообучения, позволяющий ее пополнять базу собственных знаний.

Несложно заметить, что неотъемлемой частью интеллектуальной системы являются *знания*. Знания представлять собой либо набор правил, либо иную форму представлений, позволяющую на их основе производить процедуру принятия решения в той или иной ситуации.

Знания не следует путать с данными или информацией, например, с базами данных, картами и планами местности, с данными, полученными в результате сложной обработки сенсорной информацией (например, с данными, полученными с системы технического зрения, даже если на их основе была построена локальная карта местности) и т.д. Т.е., знания определяют правила поведения системы, а вовсе не представляют собой информацию об окружающей среде или иного рода информацию. Другими словами, знания отвечают на вопрос «Что делать, если …?», а не «Что представляет собой …?»

Сами знания могут храниться в виде каких-либо данных (например, текстового файла с правилами, или записей в таблице базе данных), но сам способ хранения знаний не меняет их суть.

Вопреки ожиданиям, *интеллектуальная система сама по себе не обладает памятью*. Т.е., в обученной интеллектуальной системе, *конкретной комбинации данных, поданных на вход интеллектуальной системы, является однозначная комбинация выходных сигналов*. Если интеллектуальной системе требуется память, она должна обладать дополнительными входами и выходами, позволяющими работать с внешней памятью.

Механизм обучения или самообучения интеллектуальной системы должен позволять пользователю добавлять или изменять знания, заложенные в систему.

Если интеллектуальная система обладает механизм обучения, то данная система должна иметь редактор базы знаний, который позволяет на понятном человеку пользовательском интерфейсе формировать новые знания или изменять уже существующие.

Если интеллектуальная система обладает исключительно механизмом самообучения, то такая система должна обладать интерфейсом, позволяющим пользователю формировать обучающие примеры. Для механизмов самообучения важно правильно сформировать критерий качества работы системы, минимизация или максимизация которого дает системе возможность судить о правильности самообучения.

Следует заметить, что величины на входе интеллектуальной системы в ряде случаев представляют собой непрерывные величины, например, угол пеленга цели, или опасность того или иного направления движения. Поэтому предусмотреть все возможные комбинации входов попросту невозможно. Поэтому приходится прибегать к механизму *классификации*.

Классификация – это процесс разбиения входного значения по каждому входу на диапазоны значений. Действия системы при попадании входной величины в тот или иной диапазон может быть различно. Например, если скорость выше 60 км/ч, система совершает одни действия, при скорости от 30 км/ч до 60 км/ч система совершает другие действия, а при скорости, ниже 30 км/ч система совершает какие-либо третьи действия.

Классификация является важной составляющей работы интеллектуальной системы.

В связи со всеми перечисленными свойствами, можно выделить несколько технологий построения интеллектуальных систем:

1. Экспертные системы.
2. Фреймообразные структуры.
3. Нечеткая логика.
4. Нейронные сети.
5. Ассоциативная память.

Подробно о каждой из этих технологий речь пойдет в следующих главах.

5.2. Экспертные системы

5.2.1. Описание технологии экспертных систем

Экспертные системы – это технология интеллектуальных систем, знания в которой представлены набором продукционных правил, типа «ЕСЛИ …, ТО ….».

Например:

```
1. ЕСЛИ до_препятствия<40 И скорость>60, ТО тормоз=1
2. ЕСЛИ до_препятствия<40 И скорость>30 И
                                        скорость≤60, ТО тормоз=0.5
3. ЕСЛИ до_препятствия<40 И скорость≤30, ТО тормоз=0.1
4. ...
```

В данном примере «до_препятствия» и «скорость» – входные переменные, а «тормоз» – выходная переменная.

Обучение экспертной системы производится экспертом с помощью открытого редактора правил.

Кроме того, экспертная система может иметь механизм самообучения. Для этого экспертная система может содержать набор правил, позволяющих автоматически добавлять в базу знаний новые правила.

Процесс логического вывода в экспертной системе реализован за счет обработки продукционных правил.

Кроме того, при реализации экспертной системы можно не просто формировать выходные сигналы, но и выполнять те или иные сложные и продолжительные последовательности действий с внутренними ветвлениями и проверками.

Несложно заметить, что при большом количестве классификаций по каждому входу, количество правил экспертной системы катастрофически возрастает. Тем не менее, из-за дискретности классификации, решения, принимаемые интеллектуальной системой, также дискретны. Так, в приведенном примере, «тормоз» принимает всего 3 дискретных значения.

Достоинства экспертных систем:

- Простота реализации. Экспертная система может быть реализована на базе любого языка программирования без дополнительных библиотек и компиляторов.
- Удобство реализации системы, оперирующей с логическими или дискретными входными сигналами.
- Понятная для человеческого восприятия база знаний.
- Возможность запуска последовательности действий.

Недостатки экспертных систем:

- Сложность реализации систем, оперирующей с непрерывными сигналами. Число правил, позволяющих классифицировать входные сигналы на большое множество диапазонов в сочетании с другими входными сигналами, сильно увеличивает число правил.
- Сложность восприятия базы знаний при большом числе правил.

5.2.2. Применение экспертных систем в составе системы управления интеллектуальных мобильных роботов

Достоинства и недостатки экспертных систем, отмеченные в предыдущей главе, определяют место применения экспертных систем в общей структуре системы управления мобильных роботов.

Экспертные системы наиболее удобны в применении на стратегическом уровне системы управления (в системе управления поведением). На этом уровне системе приходится оперировать в основном с логическими и дискретными сигналами (например, признак наличия целевой точки движения, номер места назначения, наличие груза и т.п.). В то время, как принимаемое решение должно быть вполне четким и дискретным (например, номер места назначения, координаты конкретной целевой точки манипулятора или ее индекс, и т.п.).

К тому же стратегическому уровню системы управления приходится не просто принимать решение, но и формировать последовательность действий, приводящих к его реализации. В силу того, что экспертная система реализуется на обычном языке программирования, формирование последовательностей действий не составляет проблем для программиста.

На тактическом уровне системы управления также возможно применение экспертной системы, но ее база знаний будет содержать огромный набор продукционных правил, т.к. на тактическом уровне приходится оперировать в основном с непрерывными величинами (например, расстояние до целевой точки, ее пеленг, уровень опасности направления движения и т.п.). Каждый такой вход предполагает большое число классификаций, а при необходимости их сочетаний, количество правил катастрофически возрастает.

Поэтому использование экспертных систем на тактическом уровне системы управления достаточно неудобно.

На нижнем уровне системы управления применение экспертных систем также сложно по тем же причинам, что и на тактическом уровне.

5.2.3. Реализация экспертной системы в Dyn-Soft RobSim 5

Для реализации экспертной системы в Dyn-Soft RobSim 5 можно использовать блок «Виртуальный процессор на основе JavaScript». Расположение данного блока на панели блоков редактора схем и подключений бортовой ЭВМ показан на Рис. 89.

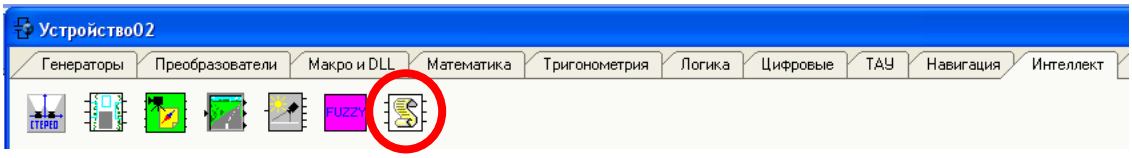

Рис. 89 Расположение блока «Виртуальный процессор на основе JavaScript» на палитре блоков редактора схем подключений бортовой ЭВМ в Dyn-Soft RobSim 5

Применение данного блока подробно описано в учебном пособии «Проектирование роботов и робототехнических систем в Dyn-Soft RobSim 5. Часть II».

С помощью данного блока пользователь может разработать программу на языке JavaScript и достаточно легко реализовать на ней экспертную систему.

Для этого в свойствах блока следует задать его входы и выходы, а также в вечном цикле реализовать набор продукционных правил, представляющих базу знаний экспертной системы (Рис. 90).

Важной частью программы на JavaScript является организация вечного цикла while, в конце которого производится синхронизация с помощью команды sleep(0).

Общая структура программного обеспечения на JavaScript, реализующая функции экспертной системы имеет следующий вид:

```
while(1)
{
   // продукционные правила
   ...
   // конец правил

   sleep(0);
}
```

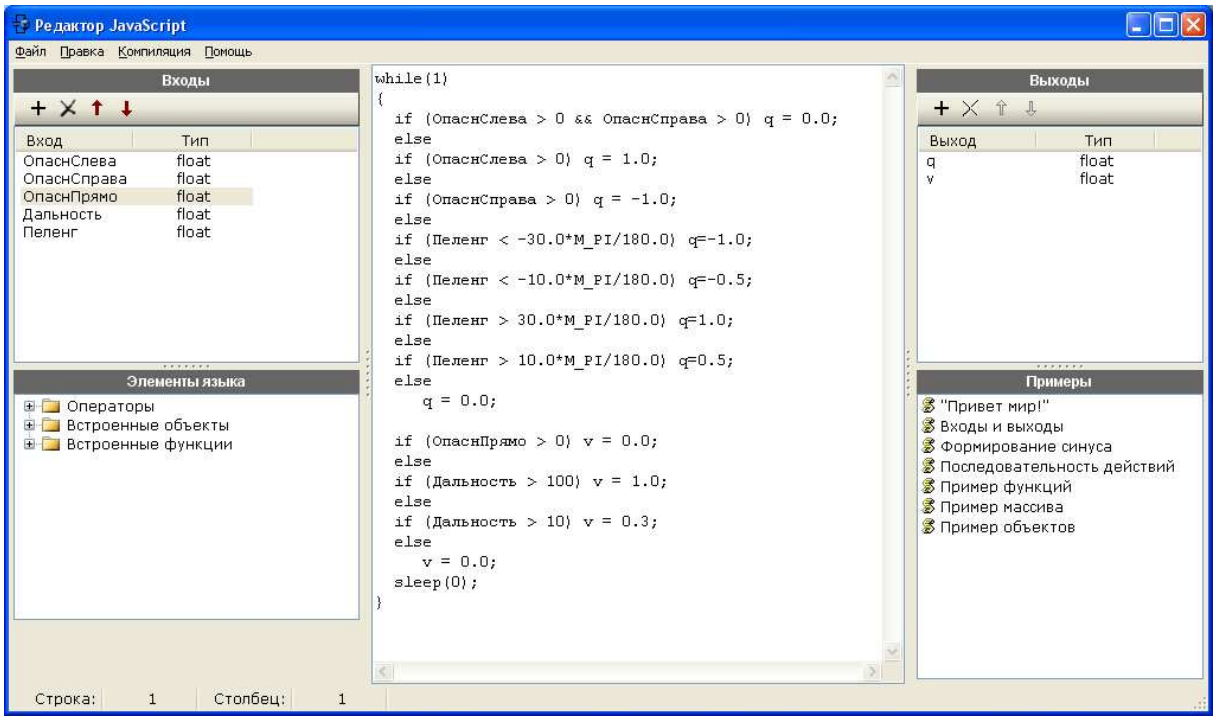

Рис. 90 Пример настройки блока «Виртуальный процессор на основе JavaScript» для реализации на нем базы знаний экспертной системы

В данном случае приведен пример абстрактной экспертной системы. Примеры экспертных систем в составе других систем робота будут показаны с следующих главах.

5.2.4. Экспертная система тактического уровня системы управления движением мобильного робота в среде с препятствиями

В данной главе приводится пример реализации экспертной системы управления автономным движением мобильного робота в среде с препятствиями.

Цель робота в приведенном примере – добраться до целевой точки, заданной координатами $X_ц$, $Y_ц$, в среде с препятствиями.

Для реализации данной системы управления интеллектуальный мобильный робот должен обладать какой-либо системой навигации (см. главу 4), возвращающей координаты робота $X_р$, $Y_р$ и угол азимута его ориентации $\alpha_р$. Также робот должен обладать системой очувствления, позволяющей определять препятствия на пути. Удобнее всего для этих целей использовать анализаторы локальной карты местности (см. главу 3.2).

Перед реализацией тактического уровня системы управления необходимо сформировать величины расстояние до цели (D) и пеленг цели α. Данные величины формируются на основании координат целевой точки $X_ц$, $Y_ц$, координат робота $X_р$, $Y_р$ и его ориентации $\alpha_р$:

$$\Delta X = X_ц - X_р$$

$$\Delta Y = Y_ц - Y_р$$

$$D = \sqrt{\Delta X^2 + \Delta Y^2}$$

$$\alpha_ц = \text{atan2}(\Delta X, \Delta Y)$$

$$\alpha = \text{lead}(\alpha_ц - \alpha_p)$$

Здесь: ΔX, ΔY – разница между координатами целевой точки и координатами робота; $\alpha_ц$ – азимут цели; atan2 – функция арктангенс 2 (частичный арктангенс); lead – функция, приводящая угол в диапазон от $-\pi$ до $+\pi$.

Данные формулы удобно реализовать в виде блоков модельно-ориентированного языка программирования Dyn-Soft RobSim 5 (Рис. 91).

В примере опускаются подробности построения системы навигации робота, а также подробности реализации алгоритмов локальной карты местности.

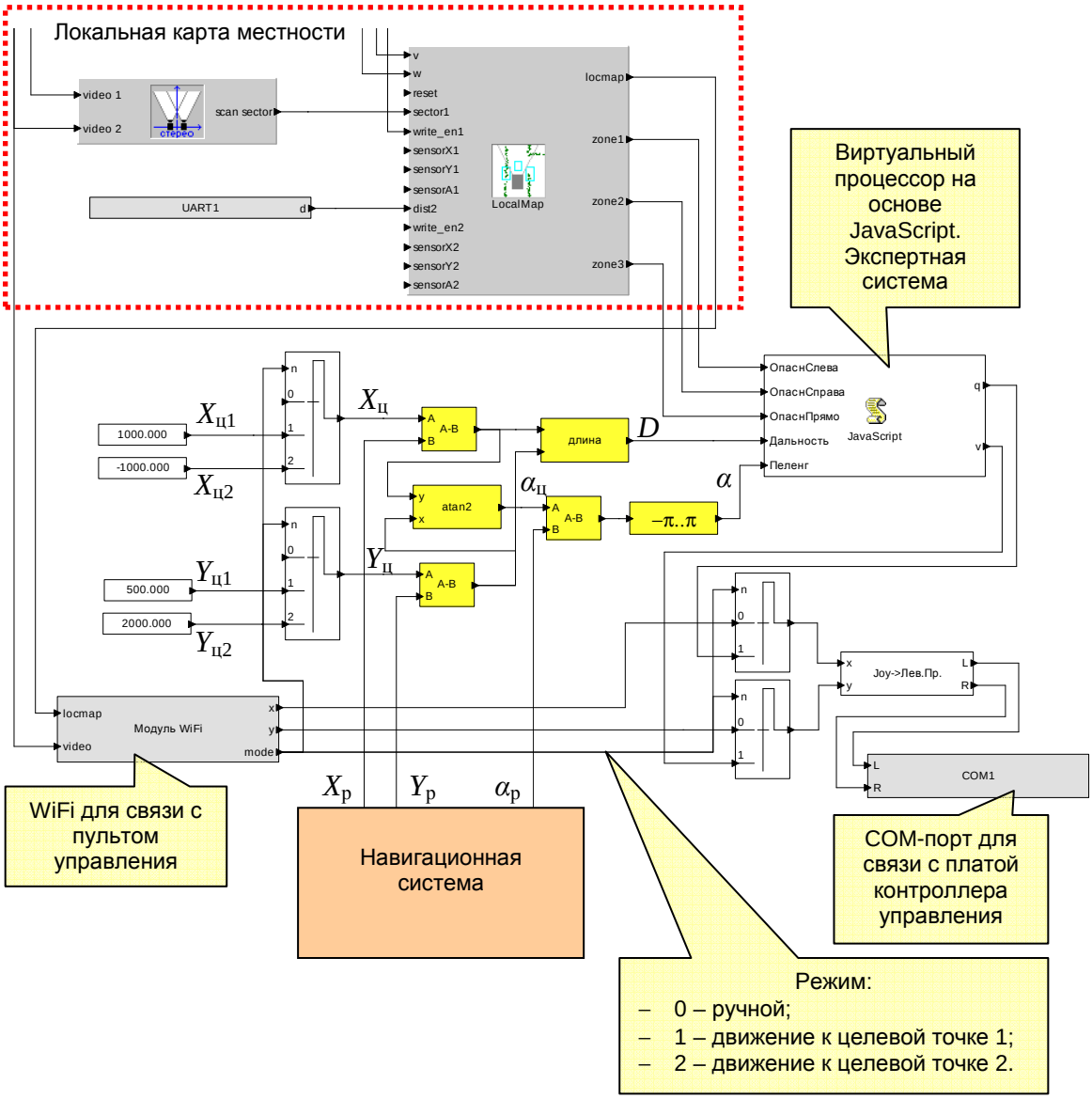

Рис. 91 Пример реализации тактического уровня интеллектуальной системы управления движением мобильного робота в среде с препятствиями на основе экспертной системы в редакторе структуры программного обеспечения Dyn-Soft RobSim 5

В примере используются две целевые точки с координатами $X_{ц1}$, $X_{ц2}$ и $X_{ц2}$, $Y_{ц2}$. Для выбора целевой точки используется сигнал *mode*, который принимает значения:

– 0 – режим ручного управления;
– 1 – режим движения к целевой точке 1;
– 2 – режим движения к целевой точке 2.

Координаты целевых точек задаются в виде констант. Для переключения активной целевой точки используются мультиплексоры, на выходе которых формируются сигналы $X_ц$ и $Y_ц$.

Группа дальнейших блоков по выше приведенным формулам формируют сигналы D и α, которые поступают на блок «Виртуальный процессор на основе JavaScript», реализующий экспертную систему.

Также на вход экспертной системы поступают данные об опасности направлений, формируемые анализаторами локальной карты местности.

На выходе экспертная система формирует сигналы q и v, соответственно скорость поворота (от -1 до +1), и скорость движения (от -1 до +1).

Данные сигналы аналогичны сигналам джойстика на пульте управления. Поэтому блок «Пересчет координат джойстика в сигнал левого и правого привода» в данной схеме устанавливается не на пульте управления, а в программном обеспечении бортовой ЭВМ. По WiFi с пульта управления поступают сигналы x, y – (float) положение рукоятки джойстика.

С помощью мультиплексоров переключается источник сигналов для блока «Пересчет координат джойстика в сигнал левого и правого привода». В режиме 0 используются координаты x, y джойстика от пульта управления, в остальных режимах – сигналы q и v от экспертной системы. Сигналы, пересчитанные с помощью блока «Пересчет координат джойстика в сигнал левого и правого привода» в скорость левого L и правого R борта передаются по порту COM1 на плату контроллера управления приводами.

Программное обеспечение блока «Виртуальный процессор на основе JavaScript» представляет следующий код, содержащий базу знаний экспертной системы:

```
while(1)
{
  if (ОпаснСлева > 0 && ОпаснСправа > 0) q = 0.0;
  else
  if (ОпаснСлева > 0) q = 1.0;
  else
  if (ОпаснСправа > 0) q = -1.0;
  else
  if (Пеленг < -30.0*M_PI/180.0) q=-1.0;
  else
  if (Пеленг < -10.0*M_PI/180.0) q=-0.5;
  else
  if (Пеленг > 30.0*M_PI/180.0) q=1.0;
  else
  if (Пеленг > 10.0*M_PI/180.0) q=0.5;
  else
     q = 0.0;
  if (ОпаснПрямо > 0) v = 0.0;
  else
  if (Дальность > 100) v = 1.0;
  else
  if (Дальность > 10) v = 0.3;
  else
     v = 0.0;
  sleep(0);
}
```

Согласно приведенным правилам, робот должен отклоняться от опасностей слева и справа, а если их нет поворачивать к цели. Также правила определяют скорость движения робота: если опасности прямо нет, то скорость формируется в зависимости от расстояния до цели. Иначе робот останавливается.

В приведенном примере была реализована простейшая база знаний интеллектуальной системы управления движением мобильного робота в среде с препятствиями.

Несложно заметить, что приведенная экспертная система формирует достаточно дискретное управление роботом. Система практически не анализирует степень опасности направлений. Также система имеет лишь две градации скорости угла поворота и скорости движения.

Для формирования более плавного управления требуется большее число правил, учитывающих различные комбинации пеленга цели и опасности направления движения.

5.2.5. Экспертная система стратегического уровня системы управления автономного мобильного робота в Dyn-Soft RobSim 5

В данной главе приведен пример реализации системы управления стратегического уровня (системы управления поведением) на базе экспертной системы в Dyn-Soft RobSim 5. Реализованная система лишь отчасти является классической реализацией экспертной системы, скорее она является сложной циклограммой со множеством ветвлений. Но собственно в этих ветвлениях и заложены знания системы.

В примере приведен интеллектуальный мобильный робот, обслуживающий клиентов, развозя им с базы какие-либо заказы. Предполагается, что робот загружается персоналом базы, а разгружается клиентами. Робот обладает звуковоспроизводящей системой, позволяющей роботу сообщать окружающим различные фразы.

Для управления роботом экспертная система имеет два выхода:

- *placeIndex* – (int) индекс целевого места назначения. Подразумевается, что тактический уровень системы управления, реализованный, например, в главе 5.2.4 или 5.4.6. Значение 0 отменяет движение. Индексы мест назначения в примере перечислены в виде констант.
- *soundIndex* – (int) номер звука для воспроизведения. Значение 0 отменяет воспроизведение. Индексы звуков перечислены в виде констант.

На входе экспертной системы следующие входы:

- *isTarget* – (bit) признак наличия целевой точки пути. Сбрасывается в 0 по приходу робота к целевой точке.

– *robotLost* – (bit) признак «Робот заблудился». Подразумевается, что тактический уровень системы управления в случае какой-либо ошибки может формировать данный признак. Например, в случае аварии или потери навигации.
– *soundPlaying* – (bit) признак воспроизведения звука. Сбрасывается в 0 по завершению воспроизведения.
– *isLoad* – (bit) признак наличия груза на роботе.

Внешний вид окна редактора языка JavaScript, редактирующего данный пример, показан на Рис. 92.

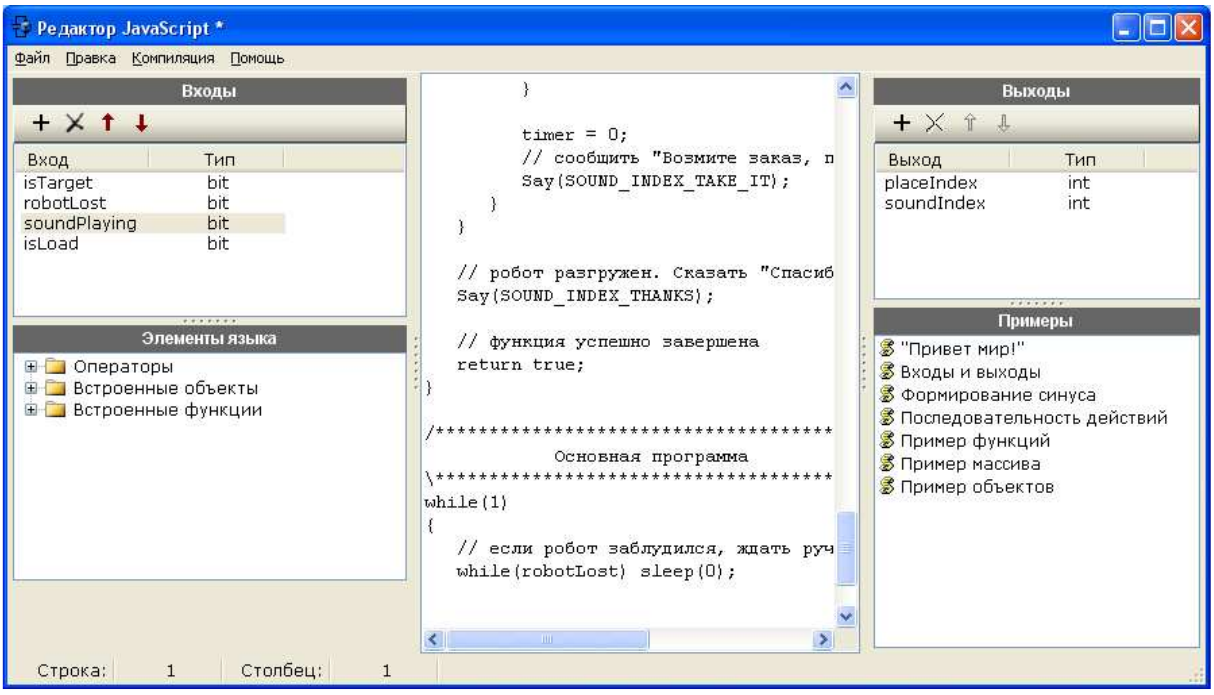

Рис. 92 Внешний вид редактора JavaScript, на базе которого реализуется стратегический уровень системы управления поведением интеллектуального мобильного робота

Текст программы на языке JavaScript для стратегического уровня системы управления поведением интеллектуального мобильного робота на базе экспертной системы, приведен ниже:

```
/************************************\
            константы
\************************************/
//---------- placeIndex ------------
PLACE_INDEX_NONE   = 0;  // нет
PLACE_INDEX_BASE   = 1;  // база
PLACE_INDEX_CLIENT = 2;  // клиент

//---------- soundIndex -------------
SOUND_INDEX_NONE        = 0; // нет
SOUND_INDEX_LOST        = 1; // фраза "я заблудился"
SOUND_INDEX_YOUR_ORDER  = 2; // фраза "Ваш заказ, пожалуйста"
```

```
SOUND_INDEX_TAKE_IT     = 3; // фраза "Возьмите заказ, пожалуйста"
SOUND_INDEX_ON_THE_BASE = 4; // фраза "Я на базе"
SOUND_INDEX_NOT_CLAIMED = 5; // фраза "Заказ не востребован"
SOUND_INDEX_IM_AWAY     = 6; // фраза "Я пошел"
SOUND_INDEX_THANKS      = 7; // фраза "Спасибо за заказ"

/*****************************************\
        функция движения к цели
  возвращает false если робот заблудился
\*****************************************/
function GoToPlace( index )
{
   placeIndex = index;  // установить цель



   // ждать окончания движения
   do
   {
      sleep(0);
      if (robotLost)
      {
         // робот заблудился, остановить робота
         placeIndex = PLACE_INDEX_NONE;
         Say(SOUND_INDEX_LOST); // сообщить "Я заблудился"
         return false; // вернуть false;
      }
   } while(isTarget);

   placeIndex = PLACE_INDEX_NONE; // остановить робота
   sleep(0);
   return true;
}

/*********************************************\
   функция говорит фразу и ожает ее окончания
\*********************************************/
function Say( index )
{
   soundIndex = index; // установить звук
   // ждать окончания воспроизведения
   do { sleep(0); } while(soundPlaying);
   soundIndex = SOUND_INDEX_NONE;
}

/*********************************************\
    Движение на базу и поведение на ней
  Вернуть false в случае, если робот заблудился
\*********************************************/
function GoToBase()
{
   // движение на базу
   if (!GoToPlace(PLACE_INDEX_BASE)) return false;

   // если робот везет груз
   if (isLoad)
   {
      // просить разгрузить робота
      while(isLoad)
      {
         // сообщить "Заказ не востребован"
```

```
         Say(SOUND_INDEX_NOT_CLAIMED);

         // ждать 10 секунд
         for(var timer=0; timer < 10000; timer+=100)
         {
            sleep(100);
            if (isLoad) break; // робот разгружен
         }
      }
   }
   else
   {
       // сообщить "Робот пришел"
       Say(SOUND_INDEX_ON_THE_BASE);
   }

   // ждать загрузки
   while(!isLoad) sleep(0);

   // сказать "Я пошел"
   Say(SOUND_INDEX_IM_AWAY);

   // функция успешно завершена
   return true;
}

/*******************************************\
    Движение к клиенту и поведение около него
  Вернуть false в случае, если робот заблудился
\*******************************************/
function GoToClient()
{
   // движение к клиенту
   if (!GoToPlace(PLACE_INDEX_CLIENT)) return false;

   // сообщить "Ваш заказ"
   Say(SOUND_INDEX_YOUR_ORDER);

   // ждать разгрузки 30 секунд
   var timer = 0;
   var n = 0;
   while(isLoad)
   {
      sleep(100);
      timer+=100;
      if (timer > 10000) // каждые 10 секунд....
      {
         if (n >= 3)
         {
            // прошло 30 секунд. Заказ не востребован
            // сообщить "Заказ не востребован"
            Say(SOUND_INDEX_NOT_CLAIMED);

            // функция успешно завершена
            return true;
         }

         timer = 0;
         // сообщить "Возмите заказ, пожалуйста"
         Say(SOUND_INDEX_TAKE_IT);
      }
```

```
   }

   // робот разгружен. Сказать "Спасибо"
   Say(SOUND_INDEX_THANKS);

   // функция успешно завершена
   return true;
}

/*****************************************\
            Основная программа
\*****************************************/
while(1)
{
   // если робот заблудился, ждать ручного вывода из этого состояния
   while(robotLost) sleep(0);

   if (!GoToBase()) continue; // идти на базу
   if (!GoToClient()) continue; // идти к клиенту
}
```

5.3.Фреймообразные структуры

5.3.1. Описание технологии фреймообразных структур

Технология фреймообразных структур – интеллектуальная технология построения алгоритмов поведения робота на основе диалога с пользователем на естественном языке.

В случае наличия у интеллектуальной системы управления большого числа разнообразных функций с разнообразными параметрами, обычный интерфейс с оператором создать сложно или невозможно. Для управления всеми функциями системы в этом случае необходимо реализовать язык текстовых или голосовых запросов.

При реализации текстовых запросов важно иметь механизм подсказок, облегчающих пользователю ввод текста. А при реализации голосовых запросов важно иметь смысловую подсказку, помогающую при разборе звука.

Для построения системы управления поведением, отвечающей всем перечисленным требованиям, можно использовать технологию фремообразных структур.

Организация базы знаний системы управления поведением на базе фреймообразных структур

Структуру базы знаний системы управления поведением представлена на Рис. 93. Она включает в свой состав *словарь* используемых слов и список *типов фреймов*.

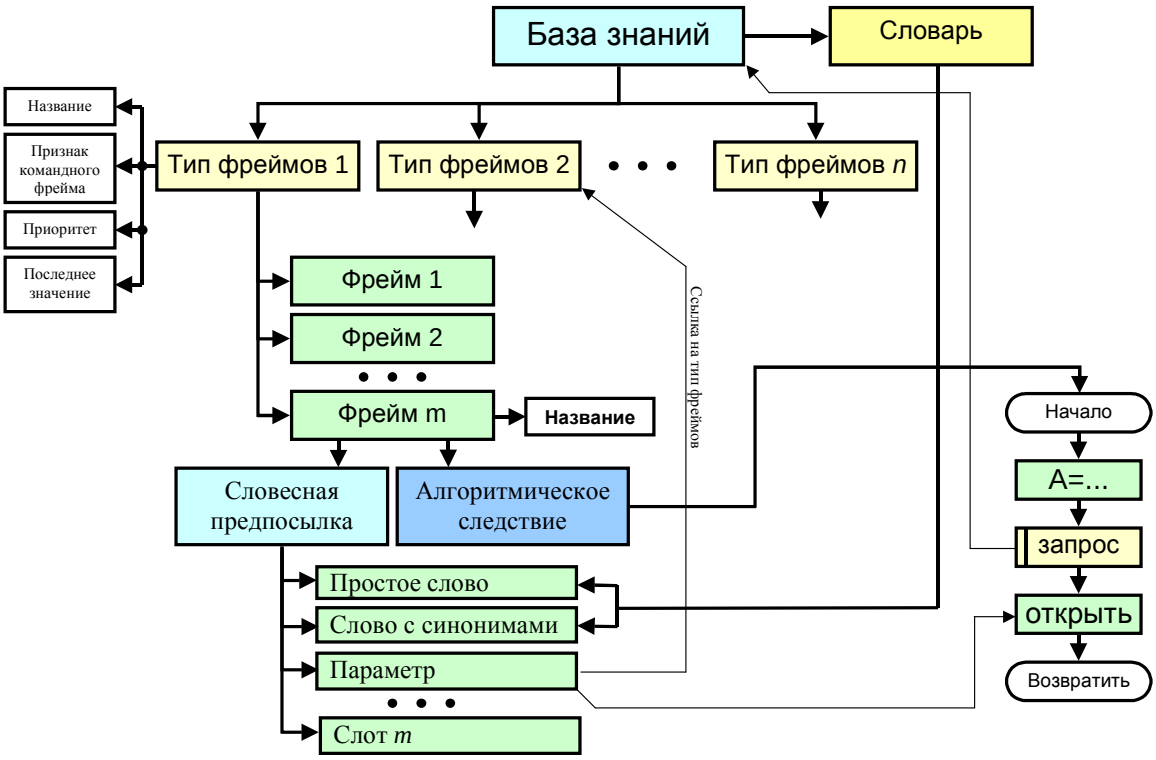

Рис. 93 Структура базы знаний системы управления поведением

Каждый *тип фреймов* содержит следующую информацию:

- Уникальное название типа фреймов. Например, «события», «задания», «объекты», «команды», «местоположения» и т.д. Эксперт, составляющий базу знаний, вправе выбирать любое название типа фреймов.
- Признак командных фреймов. Данный признак отличает фреймы, обозначающие команды, от фреймов, обозначающих описание предметной области.
- Приоритет задачи (в случае командного фрейма). Этот признак определяет приоритет, с которым алгоритмическое следствие фреймов попадает в очередь задач (см. далее). С помощью данного признака можно разделить командные фреймы на фреймы-задания, которые встают в конец очереди, и фреймы-команды, которые встают в начало очереди. Кроме того, эксперт, составляющий базу знаний, имеет возможность организовать собственную систему приоритетов.
- Последнее значение, которое вернул один из фреймов данного типа. Это актуально для раскрытия местоимений.
- Список фреймов данного типа.

Каждый фрейм из списка фреймов состоит из двух основных частей: *словесная предпосылка* и *алгоритмическое следствие*. Кроме этого, для

удобства редактирования, каждый фрейм содержит свое название, совпадающее с текстовым представлением *словесной предпосылки*.

Словесная предпосылка определяет последовательность слов, активизирующих данный фрейм. Она включает в себя список слотов. Каждый слот может являться:

- Одиночным словом. В этом случае слот хранит указатель на слово в словаре и атрибуты слота. Атрибуты слота в этом случае могут включать признак «умалчиваемое слово». Например, предложение «Возьми кубик и положи на стол» состоит из двух фраз. Вторая фраза («положи на стол») не уточняет, что именно нужно класть на стол, т.е. опускается слово «кубик». При составлении базы знаний слот может быть помечен признаком «умалчиваемое слово». В этом случае вместо данного слова может быть автоматически подставлено последнее данного типа.
- Словом и списком синонимов. В этом случае слот хранит список указателей на слова-синонимы в словаре и атрибуты слота. Атрибуты слота в этом случае могут включать признак «умалчиваемое слово».
- Параметром (ссылкой на элемент предметной области). В этом случае слот хранит название параметра, тип параметра (т.е. тип фреймов), значение по умолчанию и атрибуты слота. Значение по умолчанию актуально в случае выставленного атрибута «умалчиваемое слово». В этом случае, параметр принимает текстовое значение, указанное по умолчанию. Кроме этого атрибуты слота могут включать признак «обучаемый параметр». В этом случае на месте данного параметра в запросе может находиться любое (в том числе неизвестное) слово.

Фрейм активизируется в том случае, если в строку запроса входят все слова указанного фрейма в указанной последовательности. При этом запрос допускает наличие других слов вперемешку с искомыми. Например, фраза: «**подними** этот большой **кубик**» при прочих равных условиях активизирует фрейм со словесной предпосылкой «**подними кубик**».

Если словесная предпосылка фрейма включает слот-параметр, то поиск активизируемого фрейма производится путем подстановки вместо параметра словесных предпосылок фреймов, указанного типа.

Например, пусть имеется тип фреймов «объекты», включающий в свой состав фреймы: «кубик», «шар», «цилиндр» и т.д. Пусть имеется тип фреймов «место», включающий в свой состав фреймы: «слева от *obj*», «справа от *obj*», «перед *obj*», где *obj* – помечен, как параметр типа «объекты». Пусть имеется тип фреймов «задания», включающий в свой состав фрейм «встань *place*», где *place* – помечен, как параметр типа

«место». Тогда фраза «**встань справа от** красного **шара**» активизирует фрейм «**встань** *place*». Параметр *place* этого фрейма примет значение фрейма «**справа от** *obj*», а параметр *obj* в свою очередь примет значение фрейма «**шар**».

Алгоритмическое следствие фрейма состоит из блок-схемы алгоритма, выполняющегося при активизации фрейма. Каждый элемент блок-схемы является элементарным программным оператором. Всего таких операторов насчитывается 12 штук, среди которых:

1. Блок «Возвратить». Блок завершает алгоритм и формирует код возврата.
2. Блок «Запрос». Этот блок делает новый запрос в базу знаний, ожидает его завершения, а в указанную переменную получает код возврата последнего активизированного алгоритма. Если переменная содержит несуществующий идентификатор, то вводится новая переменная.
3. Блок «Открыть параметр». Данный блок запускает алгоритмическое следствие фрейма, переданного активному фрейму в качестве параметра из словесной предпосылки. Например, в предыдущем примере при выполнении алгоритмического следствия фрейма «встань place» данный блок может открыть параметр place, т.е. запустить алгоритмическое следствие фрейма «справа от obj».
4. Блок присвоения. Данный блок присваивает значение математического выражения (см. далее) указанной переменной или порту.
5. Блок проверки условия. Данный блок производит операцию сравнения значений двух математических выражений и в зависимости от результата сравнения производит ветвление программы. Операции сравнения может иметь одно из следующих значений: равно, больше, меньше, больше или равно, меньше или равно, не равно.
6. Блок задержки. Блок ожидает истечения заданного промежутка времени. При этом работа системы не останавливается, а просто пропускаются такты расчета.
7. Блок «Ждать». Блок ожидает выполнения указанного условия. Данный блок производит операцию сравнения значений двух математических выражений. Продолжение алгоритма осуществляется в случае, когда проверяемое условие верно, или в случае таймаута ожидания. В противном случае блок пропускает такты расчета.
8. Блок «Такт». Блок пропускает одного такта работы. Это актуально для синхронизации с внешней схемой.
9. Блок «Диалог». Данный блок ожидает от пользователя ввода нового запроса. Введенную строку блок помещает в заданную переменную.

Если задана несуществующая переменная, то в памяти появляется новая.

10. Блок «Сброс». Блок немедленно прекращает выполнение всех задач и очищается список задач (см. далее).
11. Блок «Варианты вывода». Данный блок содержит список вариантов ответа системы. Блок выбирает случайную строку из списка и помещает в выходную переменную (STDOUT), отвечающую за строку ответа пользователю.
12. Блок «Обучение». Данный блок предназначен для обучения системы новым терминам предметной области или новым действиям. Блок обладает двумя типовые настройки:
    - Обучение новым фреймам-предметам.
    - Обучение новым фреймам-действиям.

В случае обучения фреймам-предметам предполагается, что в словесной предпосылке активного фрейма существует слот-параметр, отмеченный атрибутом «обучаемый параметр». В этом случае блок «обучение» в список фреймов указанного параметром типа добавляет новый фрейм. Названию и словесной предпосылке нового фрейма присваивается слово, стоящее в запросе на месте параметра. При этом если это возможно и разрешено, это слово автоматически склоняется по числам и падежам. В качестве алгоритмического следствия новый фрейм использует блок «Возвратить» (блок 1), который в качестве возвращаемого значения использует значение математического выражения, указанного в данном блоке «обучение».

Например, пусть имеется тип фреймов «объекты», включающий в свой состав фреймы: «кубик» и «цилиндр». Пусть также имеется тип фреймов «команды», включающий в свой состав фрейм «это obj», где obj – параметр типа «объекты», отмеченный атрибутом «обучаемый параметр». Программное следствие фрейма «это obj» содержит блок «обучение». В этом блоке в качестве математического выражения используется выражение: 1+random(2000000000), т.е. случайное число от 1 до 2 000 000 000. Пусть пользователь сделал запрос: «это шар». Блок обучения в этом случае добавляет новый фрейм в список фреймов типа «объекты». Словесная предпосылка нового фрейма принимает значение: «шар» со списком синонимов: «шара», «шару», «шаром», «шаре», «шаров», «шарам», «шары», «шарами», «шарах». А алгоритмическое следствие этого фрейма генерируется из одного блока «Возвратить», формирующего код возврата 143550259.

Настройка блока обучения содержит также указания на случай, если фрейм с новым названием уже существует. В этом случае система может либо удалить старый фрейм и добавить новый, либо не добавлять новый фрейм и использовать параметры старого фрейма.

При обучении новым фреймам-действиям предполагается организация какого-либо диалога с пользователем при помощи блока, например, блока «Диалог» (блок 9). В ходе диалога пользователь вводит: а) какую-либо сложную команду; б) последовательность более простых команд, приводящих к выполнению сложной команды. Эти данные через какие-либо переменные передаются в блок «обучение». Блок «обучение» в список фреймов заданного типа добавляет новый фрейм. Названию и словесной предпосылке нового фрейма присваивается текст сложной команды. Алгоритмическое следствие нового фрейма генерируется из блока «Запрос» (блок 2) и блока «Возвратить» (блок 1). В блок «Запрос» помещается строка с последовательностью выполнения сложной команды.

Например, пусть имеется фрейм «учись», который реализует следующий диалог с пользователем:

Робот: «Введите команду для обучения».

Пользователь: «Собери пирамиду».

Робот: «Что нужно для этого сделать?»

Пользователь: «Взять кубик, поставить на другой кубик и отойти назад».

Блок «обучение» в этом случае формирует новый фрейм. Словесной предпосылке нового фрейма блок «обучение» присваивает значение «собери пирамиду». Программное следствие нового фрейма генерируется из блока «Запрос» и блока «Возвратить». В блок «Запрос» нового фрейма помещается строка запроса: «взять кубик, поставить на другой кубик и отойти назад».

Настройка блока «обучение» предполагает наличие опции, переключающей режимы генерации алгоритмического следствия. Имеется два режима: простой и расширенный. Простой режим был описан выше. В расширенном режиме строка последовательности выполнения команды разбивается на фразы. Для каждой фразы генерируется отдельный запрос, после которого проверяется результат выполнения; он должен быть ненулевым (Рис. 94). Весь алгоритм возвращает либо 0 (в случае неудачи), либо значение последнего запроса. Такая структура обеспечивает блокирование выполнение дальнейшей последовательности действий, если в одно из действий претерпело неудачу.

Настройка блока обучения содержит также указания на случай, если фрейм с новым названием уже существует. В этом случае система может либо удалить старый фрейм и добавить новый, либо игнорировать добавление нового фрейма.

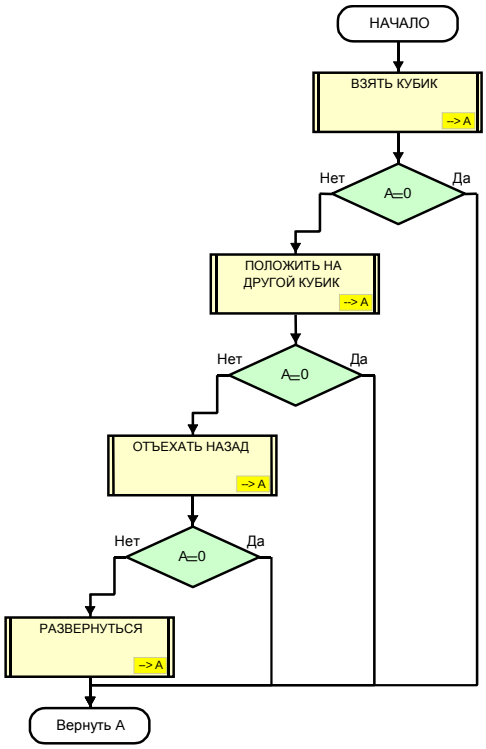

Рис. 94 Пример генерации расширенного алгоритмического следствия

При описании ряда блоков использовалось понятие *математическое выражение*. Математическое выражение представляет собой алгебраическое выражение, включающее математические и строковые функции, а также:

- Переменные или порты (входы/выходы).
- Строковое представление параметров фрейма. Здесь имеется в виду слово или выражение, стоящее в запросе на месте параметра.
- Значение параметра фрейма. В этом случае система запускает алгоритмическое следствие фрейма, переданного активному фрейму в качестве параметра из словесной предпосылки, а код его возврата используется в математическом выражении.
- Последнее значение типа фрейма.

Таким образом, алгоритмическое следствие является универсальным инструментарием для описания всех типов фреймов. Для фреймов-объектов алгоритмическое следствие состоит из одного блока «возвратить», который возвращает внутренний идентификатор объекта. Для фреймов-действий алгоритмическое следствие состоит из блоков «запрос», которые описывают последовательность выполнения сложных действий. Для описания элементарных действий алгоритмическое следствие предполагает построение блок-схемы алгоритма, который специальным образом анализирует входные переменные и формирует значение выходных переменных.

Механизм логического вывода системы управления поведением на основе фремообразных структур

Структура системы управления поведением показана на Рис. 95. Эта структура состоит из трех основных элементов: база знаний, блок обслуживания запросов и список задач.

При поступлении запроса на входе системы формируется текстовая строка, содержащая запрос. Эта строка заносится в специальную входную переменную (*STDIN*), а также поступает на блок обслуживания запроса.

Блок обслуживания запроса разделяет строку на фразы (части сложно сочиненного предложения), отбрасывает слова, начинающиеся со спецзнака, удаляет от слов частицу «-ка», после чего для каждой фразы ищет в базе знаний командный фрейм, удовлетворяющий условиям:

а) Все слова словесной предпосылки фрейма с учетом всех вложенных фреймов входят в очередную фразу в той же последовательности.

б) Общее число совпавших слов максимально.

Если этим находится хоть один такой фрейм, то блок обслуживания запроса определяет список его параметров, а также список параметров для вложенных в него фреймов.

После этого алгоритмическое следствие данного фрейма вместе со списком параметров поступают в приоритетный список задач. В зависимости от приоритета первого командного фрейма задача может оказаться либо в начале, либо в конце этого списка.

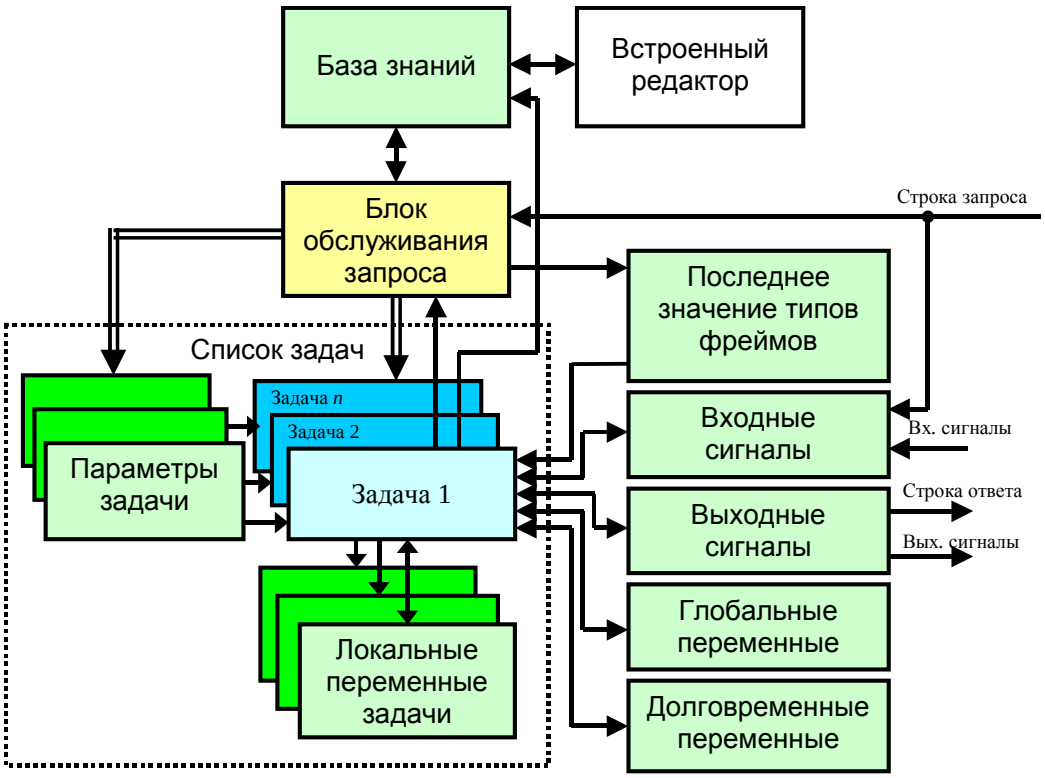

Рис. 95 Структурная схема системы управления поведением

У каждой задачи имеется список локальных переменных и параметров. Все задачи в списке задач выполняются последовательно, друг за другом. Самая первая задача является активной. После ее завершения она уничтожается из списка вместе со своими локальными переменными и параметрами, и ее место занимает следующая по списку задача.

Для чтения и записи задаче доступны:

- входные и выходные переменные (порты);
- локальные переменные задачи;
- глобальные и долговременные переменные.

Отличие глобальных от долговременных переменных заключается в том, что долговременные переменные записываются вместе с базой знаний на диск (либо другой носитель), а глобальные переменные сбрасываются при каждом запуске системы.

Только на чтение задаче доступны:

- Последние значения типов фреймов. После завершения алгоритмического следствия тип активного фрейма сохраняет возвращаемое алгоритмом значение. Это значение задача может прочитать. Актуально это для реализации местоимений.
- Текстовое представление параметров предпосылки фрейма. Если в словесной предпосылке используется параметр, то при поступлении запроса на месте параметра оказывается какое-либо слово или словосочетание. Задача имеет возможность прочитать это словосочетание.

Задача может совершать следующие действия:

- Запускать алгоритмическое следствие параметра предпосылки фрейма. При этом образуется новая задача, ставящаяся в начало списка.
- Формировать внутренние текстовые запросы, аналогичные запросу пользователя.
- Формировать новые фреймы в базе знаний.

Для редактирования содержимого базы знаний имеется встроенный редактор, позволяющий эксперту создавать и настраивать базу знаний.

При запуске системы автоматически формируется служебный запрос «ПРИ_ИНИЦИАЛИЗАЦИИ». Этот запрос активизирует одноименный служебный фрейм. По умолчанию данный фрейм не производит никаких действий, однако, у эксперта имеется возможность модифицировать этот фрейм для выполнения необходимых действий при инициализации системы.

Если блоку обслуживания запроса удается активизировать хоть один фрейм, то формируется служебный запрос «ЗАДАНИЕ_ПОНЯЛ». Данный запрос активизирует одноименный служебный фрейм, который по

умолчанию формирует один из вариантов положительного ответа пользователю, например, «ОК».

Если блоку обслуживания запроса не удается активизировать ни одного фрейма, то формируется служебный запрос «ЗАДАНИЕ_НЕ_ПОНЯЛ». Данный запрос активизирует одноименный служебный фрейм, который по умолчанию формирует один из вариантов отрицательного ответа пользователю, например, «Я Вас не понял».

Таким образом, система обладает гибкими настройками, позволяющими предпринимать какие-либо действия в случае положительного или отрицательного ответа пользователю.

Достоинства систем управления на основе фреймообразных структур являются:

- Универсальность. Она может быть настроена на любой язык (в т.ч. и метаязык событий), и может управлять любыми системами (а не только мобильными роботами).
- Наличие механизмов формирования местоимений и умалчиваемых слов.
- Наличие механизмов самообучения на основе диалога с пользователем.
- Возможность использования вложенности, как в словесных предпосылках, так и в алгоритмических следствиях.

К недостаткам следует отнести то, что система не анализирует смысл фраз, а работает по ключевым словам. Кроме того, технология фреймообразных структур может быть применена только для построения стратегического уровня системы управления роботом.

5.3.2. Пример системы управления поведением интеллектуального мобильного робота на основе фремообразных структур

В данной главе рассматривается пример реализации системы управления поведением интеллектуального мобильного робота на основе фреймообразных структур. Следует отметить, что по мере увеличения функциональных возможностей робота возрастает число возможных команд, и, как следствие, увеличивается объем базы знаний системы управления поведением. Кроме того, по мере эксплуатации робота объем базы знаний может увеличиваться за счет ручного и автоматического обучения системы.

Система управления интеллектуального мобильного робота, рассматриваемого в примере, содержит следующие подсистемы (Рис. 96):

— Система управления движением (тактический уровень системы управления). Способ реализации системы управления движением не имеет особого значения.

— Система технического зрения, позволяющая определять расстояния до объектов.
— Обучаемая база данных образов с системой идентификации образов. База данных может по команде добавлять или удалять образы из указанных пользователем координат, а также распознавать образы на изображении и выдавать их количество и координаты на местности.
— Блок расчета относительных координат. Блок позволяет рассчитывать координаты целевой точки для робота относительно координат найденной фигуры из базы данных образов. На вход поступают координаты фигуры, относительная точка x, y, а также расстояние до объекта (Рис. 97).
— нижний уровень системы управления;
— система управления поведением на основе фремообразных структур (стратегический уровень системы управления).

Система управления поведением робота обладает следующими входами:

- *STDIN* – вход запроса. На этот вход поступает строка запроса пользователя.
- Вход *Click*. Логический сигнал, возникающий после клика мышкой на видеоизображение на экране, и сохраняющийся после этого в состоянии «1» в течение 8 сек. Предназначен для организации диалога, при котором пользователь указывает мышкой на какой-либо предмет на видеоизображении и вводит команду «это *что-либо*». При этом сами координаты, указанные пользователем, для системы управления поведением не важны. Подразумевается, что эти координаты без участия системы управления поведением подаются на блок обучения образам.
- Вход *isTrg* – логический сигнал, определяющий наличие целевой точки для системы управления движением.
- Вход *FigCnt*. На этот вход поступает число обнаруженных системой технического зрения образов с заданным на выходе *FindFig* идентификатором.
- Вход *LearnRes*. На этот вход поступает код результата обучения базы данных образов, находящейся в составе системы технического зрения. Этот сигнал принимает ненулевое значение после окончания процесса обучения базы данных образов, инициированного сигналом *Reason* (см. далее).

Выходы:

1. *STDOUT* – выход строки ответа оператору.
2. *FindFig* – идентификатор образа, который следует обнаружить на видеоизображении. Данный сигнал поступает на вход модуля

поиска образов, который входит в состав СТЗ. Этот модуль на входе *FigCnt* формирует общее число найденных образов с заданным идентификатором.

3. Выходы *x*, *y* и *before*. На этих выходах формируются координаты целевой точки относительно объекта, идентификатор которого выводится на выходе *SeekFig* (Рис. 97). Значение сигналов на выходах *x* и *y* фиксируется только в момент назначения целевой точки (т.е. по переднему фронта импульса *trg* (см. ниже)). Затем в процессе движение относительное местоположение целевой точки относительно заданного объекта будет сохраняться. Сигнал *before* задает смещение целевой точки на заданную величину в направлении текущего положения робота (Рис. 97). Этот сигнал актуален и в начальный момент времени, и в процессе движения.
4. Выход *trg*. Передний фронт импульса этого сигнала устанавливает значение *сигнала о наличии целевой точки* в значение «1». Задний фронт импульса этого сигнала устанавливает значение *сигнала о наличии целевой точки* в значение «0». Этот сигнал также сбрасывается в «0» при достижении роботом целевой точки. Входной сигнал *isTrg* определяет текущее состояние этого сигнала. Кроме того, по переднему фронту этого импульса фиксируется положение целевой точки относительно заданного объекта, исходя из совокупности сигналов *x*, *y* и *SeekFig*. Полученное местоположение целевой точки относительно объекта сохраняется в течение всего времени движения робота к ней.
5. Выход *q*. Определяет направление поворота робота (-1 – влево, 1 – вправо, 0 – на месте). Поступает напрямую на приводной уровень.
6. Выход *v*. Определяет направление движения робота (-1 – назад, 1 – вперед, 0 – на месте). Поступает напрямую на приводной уровень.
7. Выход hands – выход управления наклоном захватного манипулятора.
8. Выход press – выход управления сжатием захватного устройства манипулятора.
9. Выход *Reason* – код причины добавления или удаления фигур из базы данных образов. Код 0 – значение по умолчанию. Код 1 означает «Добавить»; коды 2-8 –удалить объект с той или иной причиной (например, объект слишком широкий). В рассматриваемом примере коды причин не имеют особого значения.

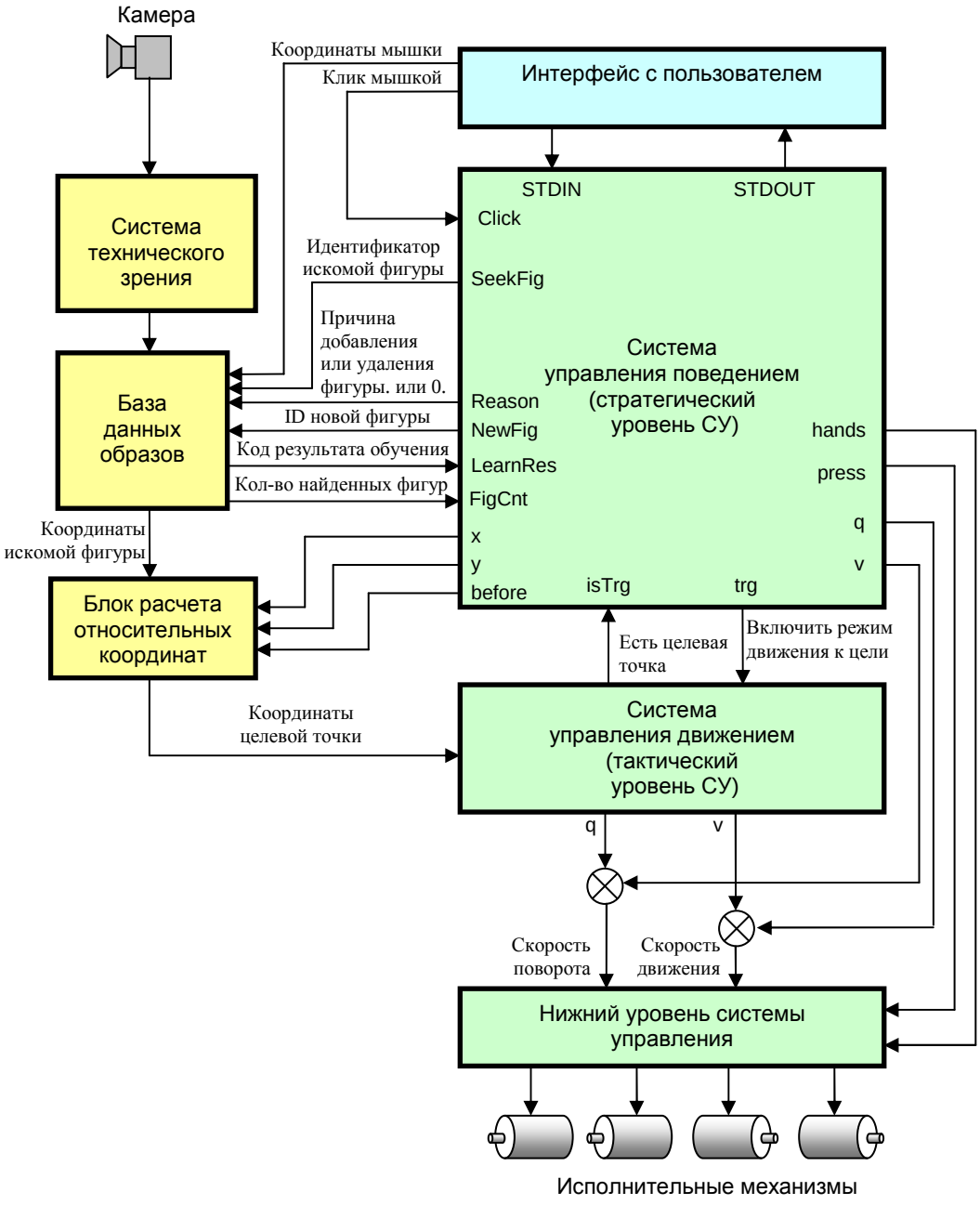

Рис. 96 Структура системы управления интеллектуальным мобильным роботом, рассматриваемом в примере

10. Выход *NewFig* – идентификатор нового объекта для добавления в базу данных образов. Значение данного сигнала актуально только в момент перехода значения сигнала *Reason* из нуля в какое-либо другое состояние. Экранные координаты нового образа формируются пользователем и поступают на блок базы данных образов без участия системы управления поведением. Актуально для реализации диалогов типа «Это что-либо».

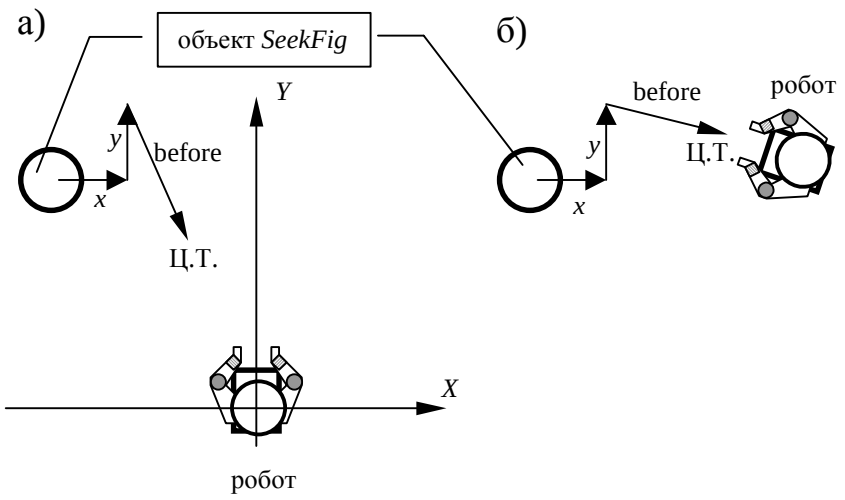

Рис. 97 Принцип назначения целевой точки (Ц.Т.) системой управления поведением: а) в начальный момент времени; б) в процессе движения к целевой точке

База знаний системы управления задается с помощью встроенного редактора. При этом автоматически формируются командный тип фреймов «события», фреймы которого обладают наивысшим приоритетом. В состав этого типа фрейма входят рассмотренные ранее фреймы: «ПРИ_ИНИЦИАЛИЗАЦИИ», «ЗАПРОС_ПОНЯЛ» и «ЗАПРОС_НЕ_ПОНЯЛ».

Для начала в базе знаний робота создаются элементарные команды. Для этого создается командный тип фреймов «задания». В данном типе фреймов создаются фреймы: «назад», «вперед», «направо», «налево». Словесная предпосылка этих фреймов будет состоять из одного единственного слота (соответственно: «назад», «вперед», «направо» и «налево»). Для фреймов «направо» и «налево» слот помечен, как слово с синонимами. Синонимами словам «налево» и «направо» являются слова «влево» и «вправо» соответственно. Алгоритмическое следствие этих четырех фреймов будет выполнять следующие действия: модифицировать выходные переменные q и v таким образом, чтобы робот начал движение в соответствующем направлении, выжидать определенный интервал времени и устанавливать нулевые значения этим переменным.

Далее создаются фреймы «езжай вперед» и «езжай назад». По смыслу данные команды должны инициировать движение вперед или назад без остановки, в отличие от команд «вперед» и «назад». Словесная предпосылка этих фреймов состоит из двух слотов. Первое слот обозначен как слово с синонимами. В качестве синонимов для слова «езжай» выступают слова «ехай» и «ехать». Для фрейма «езжай вперед» второй слот тоже обозначен как слово с синонимами. В качестве синонима слову «вперед» выступает слово «прямо». Алгоритмическое следствие этих двух фреймов модифицирует значение выходных переменных q и v таким образом, чтобы обеспечить движение вперед или назад соответственно.

Далее с помощью блока «Ждать» система ожидает модификации входной переменной *STDIN*, которая отвечает за строку запроса. Как только строка изменяется, алгоритм останавливает движение робота.

После создаются фреймы «езжай направо» и «езжай налево». Слово «езжай» содержит те же синонимы, что и в предыдущем случае. Алгоритмическое следствие этих фреймов будет выполнять следующие действия: с помощью переменных *q* и *v* и блока задержки поворачивает робота направо или налево соответственно, а затем с помощью блока «запрос» формирует запрос «езжай вперед». Таким образом, наблюдается вложенность алгоритмического следствия фреймов.

Далее вводятся знания о некоторых элементах предметной области. Пусть, база данных образов имеет возможность распознавать кубик (образ с идентификатором 1), цилиндр (образ с идентификатором 2) и шар (образ с идентификатором 3). В базе знаний системы управления поведением создается новый некомандный тип фреймов «объекты». В состав этого типа фреймов включаются фреймы: «ты», «кубик», «цилиндр» и «шар». Словесная предпосылка каждого из этих фреймов состоит из одного слота. Этот слот обозначается как слово с синонимами. В качестве синонимов выступает названия фрейма во всех падежах и числах. Например, для фрейма «кубик» список синонимов включает слова «кубика», «кубику», «кубиком», «кубике», «кубики», «кубиков», «кубиками», «кубиках». Следует отметить, что для склонения слова по падежам применяется встроенная опция редактора. Алгоритмическое следствие каждого такого фрейма состоит из одного единственного оператора «возвратить», возвращающего идентификатор объекта: для объекта «ты» – 0, для кубика – 1, для цилиндра – 2, для шара – 3.

В типе фреймов «Объекты» также создается фрейм «он». Местоимение «он» (а также «она», «оно» и т.д. во всех падежах) будет обозначать последнее значение фрейма типа «объекты». Актуально это для фраз типа «подойди к **нему**». Словесная предпосылка фрейма «он» будет содержать слово «он», обозначенное как слово с синонимами «она», «оно», «они», «его», «него», «ему», «нему», «ее», «нее», «ей», «ней», «им», «ним», «нем», «их», «них», «им», «ним», «ими», «ними». Алгоритмическое следствие фрейма «он» содержит один единственный оператор «возвратить», возвращающий последнее значение фреймов типа «объекты».

Далее необходимо описать некоторые манипуляции над объектами предметной области. Для этого в созданном ранее типе фреймов «задания» создаются следующие фреймы:

Фрейм «найди *объект*». Словесная предпосылка этого фрейма состоит из двух слотов. Первый слот «найди» обозначен, как слово с синонимами: «найти», «ищи», «искать», «поищи», «поискать». Второй слот «*объект*» обозначен, как параметр типа «объекты». Программное

следствие этого фрейма определяет алгоритм поиска объекта. Поиск осуществляется путем поворота робота на месте. Для поиска алгоритм присваивает выходной переменной *SeekFig* значение параметра «*объект*» (т.е. помещает в переменную *SeekFig* идентификатор объекта), а на входе *FigCnt* ожидает ненулевое количество найденных объектов заданного типа. Полностью алгоритм поиска приведен на Рис. 98 (б).

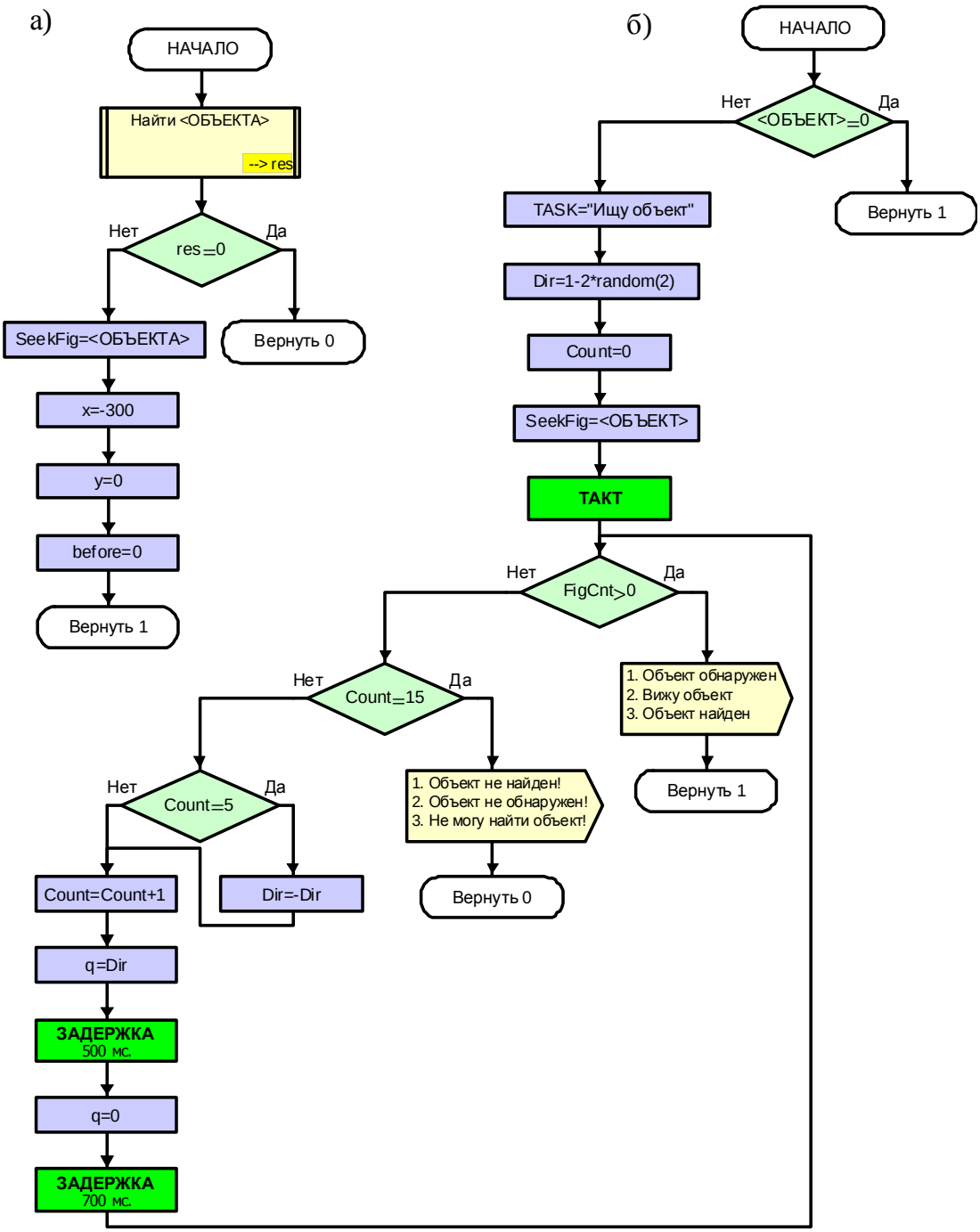

Рис. 98 Алгоритмические следствия: а) фрейма «слева от объекта»; б) фрейма «найди объект»

Фрейм «подойди к/ко *объекту*» входит в состав типа фреймов «задания». В словесной предпосылке этого фрейма слово «*объект*» является параметром типа «объекты». Алгоритмическое следствие этого фрейма производит запрос в базу данных «встань перед *объектом*». Сам запрос «встань перед объектом» будет описан далее.

Фрейм «это *что-либо*». Данный фрейм введен для реализации процессов обучения базы знаний новым объектам предметной области. Словесная предпосылка данного фрейма состоит из двух слотов. Второй слот обозначен, как параметр типа «объекты» с атрибутом «обучаемый параметр». Алгоритмическое следствие этого фрейма определяет локальную переменную *ID*, в которую записывает значение параметра «*что-либо*». В случае, когда в запросе на месте параметра стоит незнакомое слово, значение параметра будет являться пустой строкой, в противном случае – числовым идентификатором объекта. В случае пустой строки локальной переменной присваивается *ID* значение случайного идентификатора. После этого алгоритмическое следствие фрейма «это *что-либо*» реализует диалог с пользователем и предлагает пользователю показать на объект на видеоизображении. Затем алгоритм активизирует механизм обучения базы данных образов. Для этого в качестве идентификатора нового образа на выходе *NewFig* используется значение переменной *ID*. Если процесс обучения базы данных образов завершился успехом (о чем свидетельствует код результата обучения на входе *LearnRes*), то алгоритм с помощью блока «обучение» генерирует новый фрейм-объект с идентификатором *ID*. Полностью данный алгоритм приведен на Рис. 99.

Фрейм «это не *что-либо причина*». Данный фрейм предназначен для коррекции базы данных образов в случае некорректного их распознавания. Словесная предпосылка данного фрейма состоит из четырех слотов. Третий слот «*что-либо*» обозначен как параметр типа «объекты». Четвертый слот «*причина*» обозначена как параметр типа «причины удаления» и имеет атрибут «умалчиваемое слово». По умолчанию он принимает значение «БЕЗ_ПРИЧИНЫ». Алгоритмическое следствие фрейма инициирует процесс удаления образа из базы данных, причем в качестве причины удаления на выходе *Reason* алгоритм формирует значение параметра «*причина*». В случае, когда база данных образов с помощью соответствующего кода ошибки на входе *LearnRes* сообщает о том, что требуется уточнить причину, алгоритм организует диалог с пользователем, в результате которого выясняет причину. Эта причина сохранятся в локальную переменную *D*, а затем формируется с помощью блока «запрос» формируется запрос «Это не <*что-либо*> потому что [*D*]». При этом конструкция <*что-либо*> заменяется на значение параметра «*что-либо*», а конструкция [*D*] заменяется на значение локальной

переменной *D*. Полностью алгоритмическое следствие данного фрейма приведено на Рис. 100.

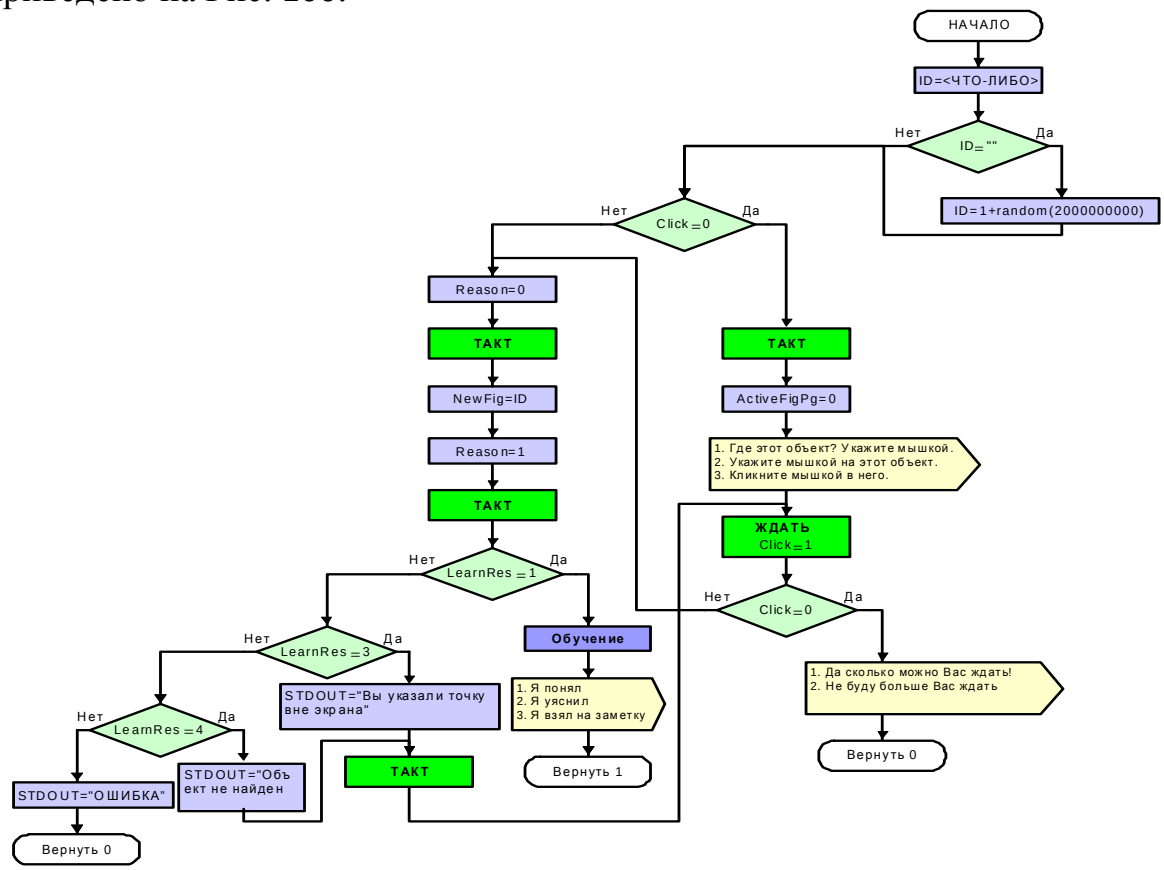

Рис. 99 Алгоритмическое следствие фрейма «это *что-либо*»

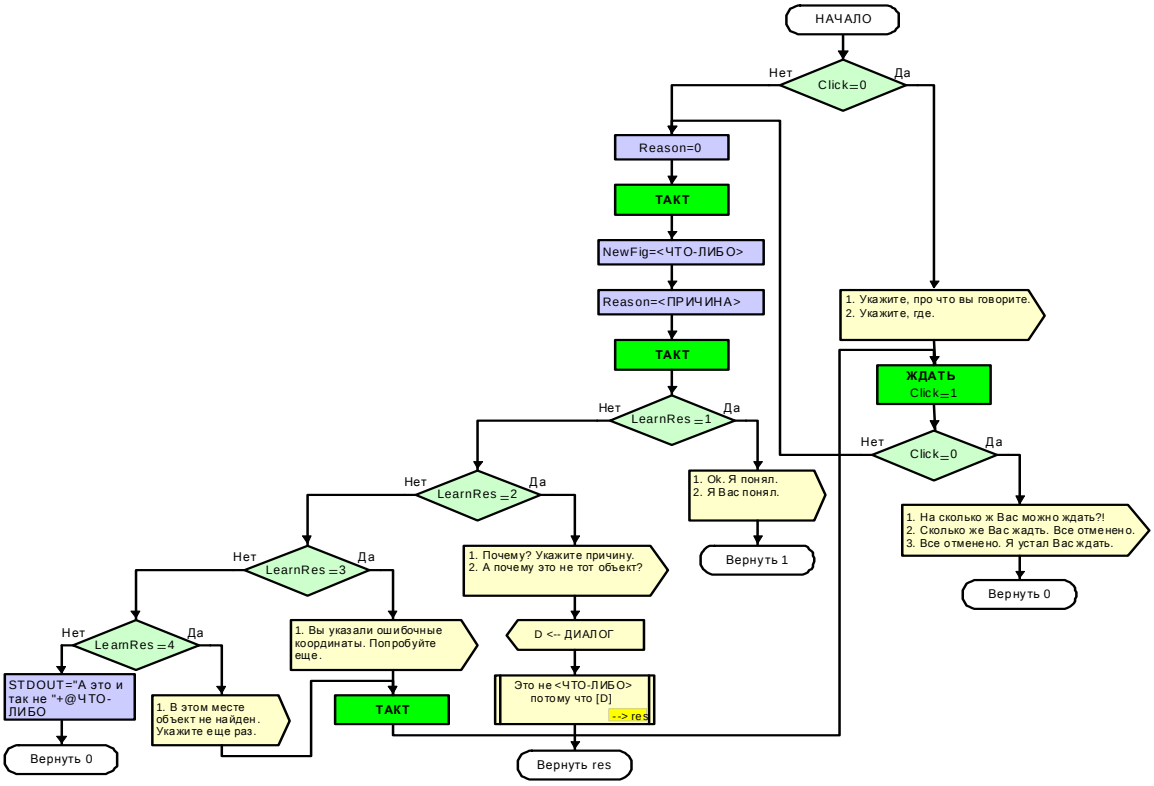

Рис. 100 Алгоритмическое следствие фрейма «это не *что-либо причина*»

Фрейм «обучение». Данный фрейм активизируется по ключевому слову «обучение». Он предназначен для обучения базы знаний системы управления поведением новым действиям. Алгоритмическое следствие этого типа организует диалог с пользователем, в результате которого выясняет сложную команду и последовательность ее выполнения. После чего с помощью блока «обучение» генерирует в базе данных новый фрейм типа «задание». Описание механизма этого обучения было приведено в главе 5.3.1.

Фрейм «встань *куда-либо*». Словесная предпосылка этого фрейма состоит из двух слотов. Первый слот – слово с синонимами «встань», «встать», «переместись», «переместиться». Второй слот – параметр типа «место» (см. далее). Алгоритмическое следствие этого фрейма открывает значение параметра «*куда-либо*» (т.е. запускает алгоритм фрейма, переданного в качестве параметров). Если алгоритм этого параметра увенчался успехом, то выходной переменной *trg* присваивается значение 1, тем самым инициируется процесс начала движения, после в течение некоторого времени ожидается выполнение условия *isTrg*=0, что будет соответствовать приходу робота в заданную целевую точку, а затем переменной *trg* присваивается значение 0.

Далее вводится некомандный тип фреймов «причины удаления», в котором фреймами перечислим причины удаления, необходимые для фрейма «это не *что-либо причина*». Словесная предпосылка этих фреймов содержит ключевое слово причины удаления. Алгоритмическое следствие этих фреймов содержит единственный блок «возвратить», который определяет код причины удаления. Важно наличие фрейма «БЕЗ_ПРИЧИНЫ», т.к. именно это значение фрейм «это не *что-либо причина*» использует по умолчанию в качестве параметра «*причина*».

Для робота определяется понятие места, актуальность которого диктует наличие таких фреймов, как «встань *куда-либо*». Для этого вводится некомандный тип фреймов «место».

Для этого типа фреймов определяются фреймы: «перед *объектом*», «слева от/о *объекта*», «с левой стороны *объекта*», «справа от/ото *объекта*», «справой стороны *объекта*», «за *объектом*», «с передней стороны *объекта*», «с задней стороны *объекта*», «у/около *объекта*». Во всех этих фреймах слово «объект» выступает в качестве параметра типа «объекты». Программное следствие этих фреймов делает запрос в базу знаний «найди объект». Если при этом процесс поиска увенчается удачей, то соответствующим образом модифицируются выходные переменные *SeekFig*, *x*, *y* и *before*. Заметим, что алгоритм не инициирует начало движения, а лишь модифицирует указанные переменные. Полностью алгоритмическое следствие данного фрейма показано на Рис. 98 (а).

Аналогичным образом в базу знаний мобильного робота были внесены прочие фреймы. Полный список фреймов базы знаний показан в Табл. 1.

Табл. 1 Список фреймов системы управления поведением ИМР, оснащенного захватным устройством

тип **События** *(командные, приор.200)*
- ПРИ_ИНИЦИАЛИЗАЦИИ
- ЗАПРОС_ПОНЯЛ
- ЗАПРОС_НЕ_ПОНЯЛ

тип **Задания** *(командные, приор. 1)*
- НАЙДИ *ОБЪЕКТ* (объекты)
- ПОДОЙДИ К *ОБЪЕКТУ* (объекты)
- ВСТАНЬ *КУДА-ЛИБО* (место)
- ЭТО *ЧТО-ЛИБО* (объекты)
- ЭТО НЕ *ЧТО-ЛИБО* (объекты) *ПРИЧИНА* (причины удаления)
- ОБУЧЕНИЕ ДЕЛАТЬ
- ВПЕРЕД
- НАЗАД
- НАЛЕВО
- НАПРАВО
- КРУГОМ
- ЕЗЖАЙ ВПЕРЕД
- ЕЗЖАЙ НАЗАД
- ЕЗЖАЙ НАПРАВО
- ЕЗЖАЙ НАЛЕВО
- НАКЛОНИСЬ ВПЕРЕД
- РАЗОГНИСЬ
- ВЫГНИСЬ НАЗАД
- НАКЛОНИСЬ НАЗАД
- СОЖМИ РУКИ
- РАЗОЖМИ РУКИ
- ОТПУСТИ
- ВОЗЬМИ *ОБЪЕКТ* (объекты)
- ПОЛОЖИ *ОБЪЕКТ* (объекты)
- СЛЕДИ ЗА *ОБЪЕКТОМ* (объекты)
- ОТНЕСИ *ЧТО* (объекты) *КУДА* (место)
- ПОЛОЖИ НА ПОЛ

тип **Объекты** *(некомандные)*
- ТЫ
- ОН
- КУБИК
- ЦИЛИНДР
- КРУГ

тип **Команды** *(командные, приор.2)*
- ВИДИШЬ ТЫ *ЧТО-ЛИБО* (объекты)
- ЧТО ДЕЛАЕШЬ
- ЧЕМ ЗАНЯТ
- СКОЛЬКО ТЫ ВИДИШЬ *ОБЪЕКТОВ* (объекты)
- СКОЛЬКО *ОБЪЕКТОВ* (объекты) ТЫ ВИДИШЬ
- СТОП

тип **Место**
- ПЕРЕД *ОБЪЕКТОМ* (объекты)
- СЛЕВА ОТ *ОБЪЕКТА* (объекты)
- С ЛЕВОЙ СТОРОНЫ *ОБЪЕКТА* (объекты)
- СПРАВА ОТ *ОБЪЕКТА* (объекты)
- С ПРАВОЙ СТОРОНЫ *ОБЪЕКТА* (объекты)
- ЗА *ОБЪЕКТОМ* (объекты)
- С ПЕРЕДНЕЙ СТОРОНЫ *ОБЪЕКТА* (объекты)
- С ЗАДНЕЙ СТОРОНЫ *ОБЪЕКТА* (объекты)
- ЧУТЬ ЛЕВЕЕ *ОБЪЕКТА* (объекты)
- ЧУТЬ ПРАВЕЕ *ОБЪЕКТА* (объекты)
- У/К *ОБЪЕКТА* (объекты)

НА *ОБЪЕКТ*
объекты

тип **Причины удаления** *(неком.)*
БОЛЬШОЙ
ФОРМА
ШИРОКИЙ
УЗКИЙ
МАЛЕНЬКИЙ
ЦВЕТ
БЕЗ_ПРИЧИНЫ

Условное обозначение:

НАЗАД – одиночное слово или слово с синонимами;

У/К – слово с явно указанными синонимами;

ОБЪЕКТ – слово, отмеченное как параметр указанного типа;
объекты

ОБЪЕКТ – слово, отмеченное как параметр указанного типа, причем слово может являться неизвестным словом;
объекты

ПРИЧИНА – параметр может отсутствовать;
причины удаления

ОТ – слово может отсутствовать.

В качестве примера для данного робота был поставлен эксперимент по сборке предметов вместе, причем действовать робот должен в среде с препятствиями.

Пусть робот первоначально находится в среде с препятствиями и расположен так, как показано на Рис. 101 (а, б).

Оператор делает запрос к роботу: «Робот, **отнеси**-ка один **кубик** к другому **кубику**».

Фраза разлагается на фразы «Робот» и «**Отнеси**-ка один **кубик** к другому **кубику**» (частичка «-ка» автоматически удаляется). Фраза «Робот» не находит эквивалента в базе знаний. Фраза «**Отнеси** один **кубик** к другому **кубику**» в базе знаний активизирует следующий командный фрейм:

ОТНЕСИ *ЧТО* *КУДА*
объекты место

При этом параметр «ЧТО» равен «кубик», а параметр «КУДА» равен «к другому кубику».

Согласно алгоритмическому следствию этого фрейма необходимо проверить, что в данный момент находится в руках у робота (идентификатор объекта, который робот держит в руках, хранится в глобальной переменной $Hold; если $Hold=0, то робот ничего не держит в руках). Если в руках у робота объект, идентификатор которого не совпадает с идентификатором объекта, который хранится в параметре «ЧТО», то алгоритмическое следствие фрейма делает запрос «Взять <ЧТО>». В данном случае у робота нет ничего изначально в руках, поэтому $Hold=0. Идентификатор объекта «кубик» определяется путем вызова соответствующего фрейма, который возвращает идентификатор объекта, равный 1. Т.к. 1≠$Hold, производится запрос «Взять кубик».

Фраза «взять кубик» активизирует фрейм:

ВОЗЬМИ *ОБЪЕКТ*
объекты

У данного фрейма один из синонимов слова «ВОЗЬМИ» является словом «ВЗЯТЬ». Согласно алгоритмическому следствию этого фрейма сначала необходимо проверить, что в данный момент в руках у робота. Если у робота что-то есть в руках ($Hold≠0), то вызывается фрейм: «положи <*ОБЪЕКТ*>». В данном случае у робота нет ничего в руках. Далее алгоритмическое следствие фрейма «возьми объект» делает запрос «встань перед <*ОБЪЕКТ*>». В данном случае «встань перед **кубик**».

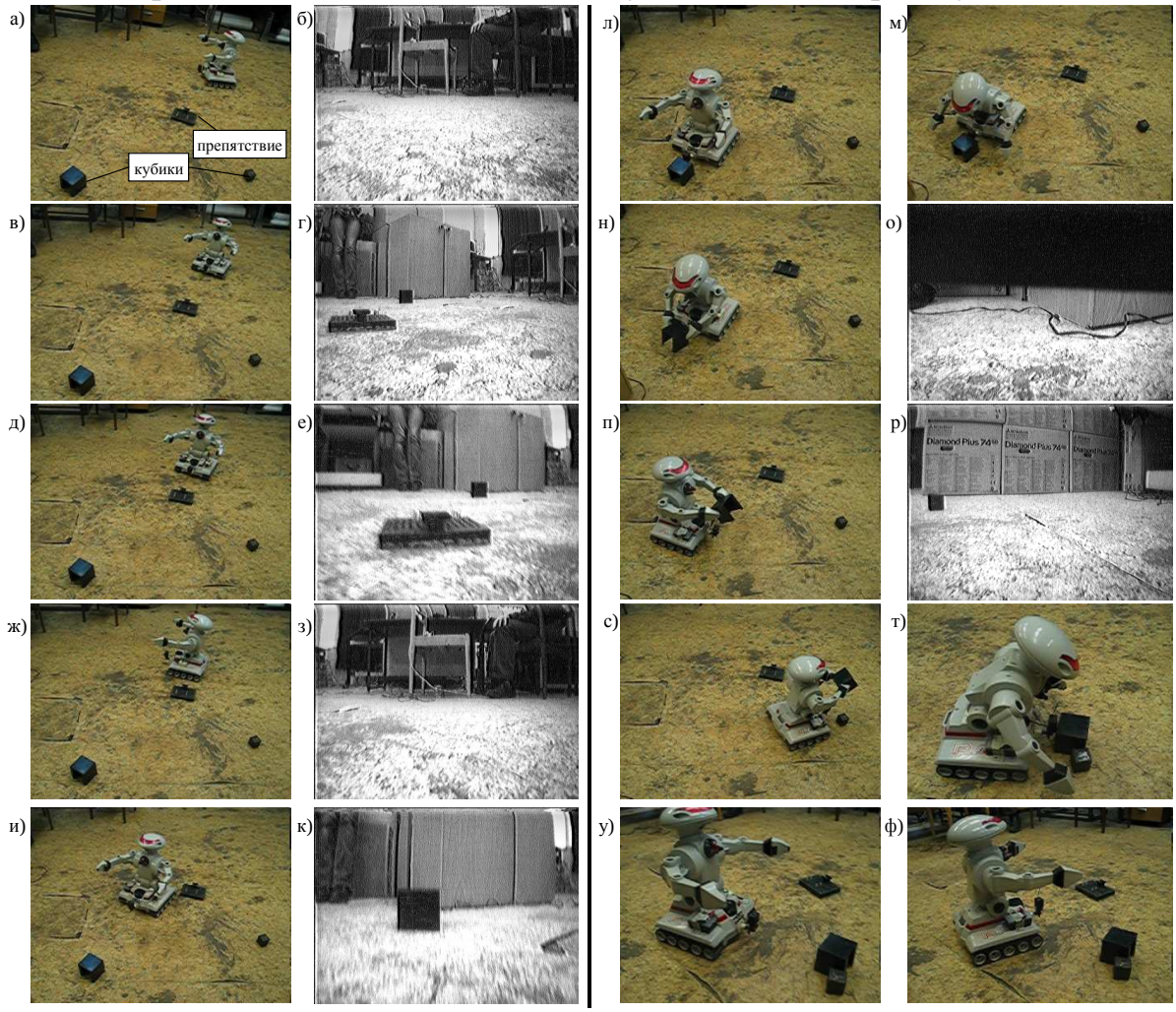

Рис. 101 Этапы выполнения роботом команды «Робот, отнеси-ка один кубик к другому кубику»

Фраза «встань перед **кубик**» активизирует фрейм:

ВСТАНЬ *КУДА-ЛИБО*
место

Здесь в качестве параметра «КУДА-ЛИБО» выступает словосочетание «перед **кубик**». Согласно алгоритмическому следствию этого фрейма необходимо раскрыть понятие «КУДА-ЛИБО». При этом вызывается алгоритмическое следствие фрейма:

ПЕРЕД *ОБЪЕКТОМ*
объекты

Здесь в качестве параметра «*ОБЪЕКТОМ*» выступает слово «кубик». Согласно алгоритмическому следствию этого фрейма необходимо сначала найти объект. Для этого выполняется запрос «найти <*ОБЪЕКТОМ*>». В данном случае это запрос «найти **кубик**».

Запрос «найти **кубик**» активизирует фрейм:

НАЙДИ *ОБЪЕКТ*
объекты

В качестве параметра «*ОБЪЕКТ*» здесь выступает слово «кубик». Алгоритмическое следствие этого фрейма раскрывает понятие «кубик» путем вызова алгоритмического следствия соответствующего фрейма. Алгоритмическое следствие фрейма «кубик» возвращает идентификатор объекта «кубик» равный 1. Этот идентификатор выводится через выходную переменную *FindFig* на базу данных объектов. Если образ не найден, то алгоритмическое следствие фрейма «найди объект» начинает поворачивать робота вправо или влево (определяется случайным образом). В данном случае это было левое направление. После поворота влево на видеоизображении был обнаружен объект, похожий на кубик (Рис. 101, б, в). В этом случае для оператора формируется ответ «Объект обнаружен», а выполнение алгоритмического следствия «найди объект» заканчивается с кодом возврата 1 (удача).

После продолжается выполнение фрейма «перед объектом». Согласно алгоритмическому следствию этого фрейма формируется значение выходных сигналов *x*, *y*, *before* и *SeekFig* таким образом, чтобы задать точку перед объектом «кубик»: *x*=0; *y*=0; *before*=200; *SeekFig*=1. Затем алгоритмическое следствие этого фрейма заканчивается с кодом возврата 1 (удача).

Далее продолжает работать алгоритмическое следствие фрейма «встань куда-либо». Оно выставляет сигнал *trg*=1. При этом выставляется сигнал о наличии целевой точки. Параметры же целевой точки были выставлены фреймом «перед объектом». Робот начинает движение к указанной целевой точке; это движение осуществляется средствами системы управления движением. Процесс движения проиллюстрирован на Рис. 101 (д-л). После прихода робота к целевой точке выполнение фрейма «встань перед объектом» заканчивается с кодом 1 (удача).

Управление переходит обратно фрейму «взять объект». Алгоритмическое следствие этого фрейма делает запрос «согнуться, сжать руки, разогнуться». При этом вызываются соответствующие фреймы. При этом робот нагибается, берет объект и разгибается (Рис. 101, м-о). Выполнение фрейма «взять объект» на этом заканчивается, а управление переходит обратно к фрейму «отнеси что куда».

Алгоритмическое следствие этого фрейма делает запрос «встань <куда>». В данном случае «встань перед **кубику**». Работа этого запроса

была рассмотрена выше. Робот производит поиск кубика (Рис. 101, п, р) и подходит к нему (Рис. 101, с). Далее фрейм «отнеси что куда» делает запрос «согнуться, разжать руки, разогнуться, назад, назад, влево». При этом робот нагибается, разжимает руки Рис. 101 (т), разгибается, отъезжает назад Рис. 101 (у) и поворачивается влево Рис. 101 (ф). На этом выполнение всех задач заканчивается.

Таким образом, сформированная база знаний системы управления поведением обладает следующими свойствами:

1. Наличие команд низкого уровня. Это позволяет делать запросы типа «отъехать вперед», «отступи назад», «повернись направо», «найди шар» и т.д.
2. Наличие команд высокого уровня, которые позволяют делать запросы типа «встань слева от шара», «подъехать к шару» и т.д.
3. Наличие механизмов обучения новым объектам. Это позволяет делать запросы типа «это апельсин», «а это не апельсин, потому что он маленький».
4. Наличие механизмов обучения новым действиям. Это позволяет организовывать следующие диалоги:

   Пользователь: начинаю **обучение**.
   Робот: Введите команду.
   Пользователь: провести инспекцию.
   Робот: что я должен сделать?
   Пользователь: подъехать к кубику, затем встать слева от шара, развернуться и подъехать к цилиндру.
   Робот: я понял.

Разработанная система управления поведением и ее база знаний были успешно апробированы на экспериментальном образце интеллектуального мини-робота. Они показали свою высокую эффективность, надежность и быстродействие.

5.3.3. Применение фреймообразных структур в Dyn-Soft RobSim 5

Фреймообразные структуры в настоящей версии Dyn-Soft RobSim 5 не представлены. Однако пользователь при желании может разработать собственный плагин для Dyn-Soft RobSim 5, и реализовать на его базе работу с фреймообразными структурами.

Примеры реализации плагинов на языке C++ и Delphi находятся в папке plugins программного комплекса Dyn-Soft RobSim 5.

5.4.Нечеткая логика

5.4.1. Описание технологии нечеткой логики

Нечеткая логика – (fuzzy logic) интеллектуальная технология, основанная на нечетких множествах Лофти Заде.

В нечетких логических системах, также как в экспертных системах, имеется набор продукционных правил, типа «ЕСЛИ …, ТО…». Но в отличие от экспертных систем результат вычисления левой части выражения «ЕСЛИ…» может принимать не только значения ПРАВДА и ЛОЖЬ, но и промежуточные значения, например значение 0,75. Результат логического вывода в нечетких логических системах также нечеткий и определяется средневзвешенным значением.

Нечеткая логическая система имеет набор входов и выходов, называемых *входными* и *выходными переменными*. Каждой комбинации входных значений в нечетной логической системе соответствует однозначная комбинация выходных значений. Т.е. нечеткая логическая система работает как конечный автомат без памяти.

По каждой входной переменной в нечеткой логической системе создается набор *функций принадлежности*, определяющих принадлежность конкретного значения входной переменной к тому или иному понятию, называемому *терму*.

Например, путь нечеткая логическая система имеет вход расстояния до цели *D*. По данному входу создается набор функций принадлежности, определяющих принадлежность конкретного значения расстояния к понятиям «Далеко» и «Близко» (Рис. 102).

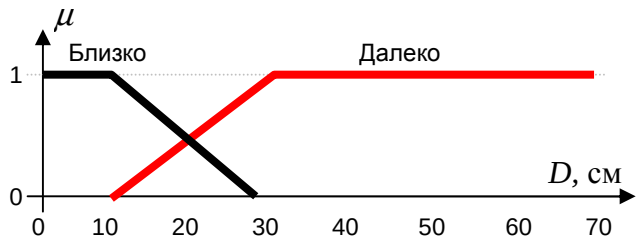

Рис. 102 Пример функций принадлежности по входу скорости

Функция принадлежности составляется на основе экспертной оценки. Так, например, на основе экспертной оценки было получено, что расстояние более 30 см – это далеко. Расстояние 20 см к понятию «Далеко» относится на 50%, расстояние 10 см к понятию «Далеко» уже не относится. Таким образом, составляется функция принадлежности терма «Далеко». Аналогичным образом составляется функция терма «Близко».

Для обеспечения плавного перехода системы из режима «Далеко» в режим «Близко» функции принадлежности делают пересекающимися.

Число термов по входным переменным определяется необходимым количеством классификаций. В ряде случаев для нечеткой логики достаточно двух классификаций.

Выходные переменные также разбиваются на термы, по которым создаются функции принадлежности. Например, по выходной переменной *u* (управление), которая изменяется от -1 до +1, определяются термы «Вперед» и «Тормоз». Функции принадлежности этих термов показаны на Рис. 103.

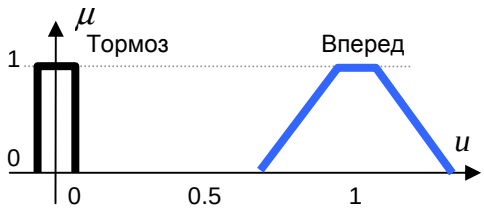

Рис. 103 Пример функций принадлежности по выходной переменной *u* (управление)

Вопреки утверждениям некоторых теоретиков, наличие покрытия функциями принадлежности всего диапазона выходных значений вовсе не обязательно.

Кроме функций принадлежности в нечеткой логической системе существует база продукционные правила, типа «ЕСЛИ …, ТО …». Операндами этих правил служат термы. Под оператором «ЕСЛИ» могут присутствовать только входные термы, а под оператором «ТО» – только выходные. Например:

```
ЕСЛИ Далеко ТО Вперед
ЕСЛИ Близко ТО Тормоз
```

Под оператором «ЕСЛИ» можно комбинировать несколько термов, объединяя их по «И», «ИЛИ» или «НЕ». У операций имеется приоритет. Операция «НЕ» имеет наивысший приоритет, а приоритет операции «И» выше, чем у операции «ИЛИ». Для изменения приоритетов операций следует использовать круглые скобки.

В правых частях выражения можно активировать несколько выходных термов, объединяя их по «И». Например:

```
ЕСЛИ Далеко И НЕ Опасно ТО Вперед И Зеленый
ЕСЛИ Близко ИЛИ Опасно ТО Тормоз И Красный
```

Здесь: «Опасно» – терм по входной переменной «Опасность», (функция принадлежности терма непредставлена в примере); «Зеленый» и «Красный» – термы по выходной переменной «Цвет лампы» (функция принадлежности терма также непредставлена в примере).

Порядок работы нечеткой логической системы следующий:

1. Фаззификация. По графикам функций принадлежности определяется достоверность каждого терма. Данная достоверность изменяется от 0 до 1.
2. Нечеткий логический вывод. Последовательно выполняются продукционные правила. При этом определяется достоверность левых частей правил по следующему принципу: операция «И» выбирает минимальное значение достоверности термов-операндов. Операция «ИЛИ» выбирает максимальное значение термов-операндов. Операция «НЕ» рассчитывает достоверность по формуле 1-α, где α – достоверность операнда операции «НЕ». Полученная достоверность левой части правила применяется для модификации правой его части, включая все перечисленные через «И» выходные термы. Причем существует два метода нечетного вывода: MAX-MIN и MAX-DOT. При методе MAX-MIN выходные функции принадлежности усекаются сверху значением достоверности левой части правила. При методе MAX-DOT выходные функции принадлежности масштабируются до этого значения достоверности. Если одна и та же выходная функция принадлежности активируется несколькими правилами, то выбирается максимальное значение достоверности.
3. Дефаззификация. Процедура дефаззификации заключается в переводе нечеткого значения выходных функций принадлежности в численные значения выходов. Для этого по каждой выходной переменной определяется центр масс фигуры, образованной модифицированными в результате нечеткого логического вывода функциями принадлежности по следующей формуле:

$$y_{\text{вых},j} = \frac{\int_{y_{\min,j}}^{y_{\max,j}} f_j(y) \cdot y \, \mathrm{d}y}{\int_{y_{\min,j}}^{y_{\max,j}} f_j(y) \, \mathrm{d}y}$$

Где: $y_{\text{вых},j}$ – численное значение j-ого выхода; $f_j(y)$ – функция, определяющая контуры результирующей фигуры, образованной модифицированными функциями принадлежности; $y_{\min,j}$, $y_{\max,j}$ – диапазон определения переменной $y_{\text{вых},j}$.

Например, пусть имеется нечеткая логическая система с входными функциями принадлежности, показанными на Рис. 102, и выходными,

показанными на Рис. 103. Продукционные правила нечетной логической системы будут следующими

```
1. ЕСЛИ Далеко ТО Вперед
2. ЕСЛИ Близко ТО Тормоз
```

Метод логического вывода – MAX-MIN.

Пусть входное расстояние в один из моментов времени будет 27 см.

Нечетная логическая система на первом этапе производит фаззификацию, т.е. определяется достоверность входных термов. В данном случае достоверность терма «Далеко» будет 0.85, а терма «Близко» – 0.15 (Рис. 104).

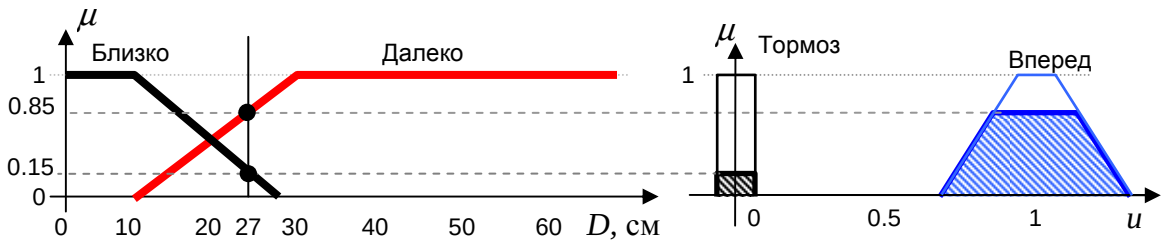

Рис. 104 Пример модификации выходных функций принадлежности методом MAX-MIN

На втором этапе нечеткая логическая система переходит к выполнению продукционных правил.

Достоверность левой части правила 1 равняется 0.85. Согласно данному правилу активизируется функция принадлежности выходного терма «Вперед». При этом согласно методу MAX-MIN выходная функция принадлежности отсекается значением 0.85 (Рис. 104).

Аналогичным образом по правилу 2 функция принадлежности терма «Тормоз» отсекается значением 0.15.

На третьем этапе нечеткая логическая система производит дефаззицикацию. При этом определяется центр масс по всем результирующим фигурам, получившимся по выходным переменным. В данном случае находится центр масс фигуры, изображенной на Рис. 105. Из рисунка видно, что центр масс приходится на значение 0.76. Данное значение и будет являться выходным значением нечетной логической системы.

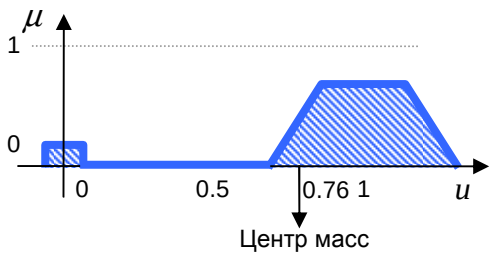

Рис. 105 Пример дефаззификации результирующей функции принадлежности

5.4.2. Достоинства и недостатки нечетких логических систем

По сравнению с экспертными системами, нечеткие логические системы обладают рядом достоинств:

- Возможность принятия средневзвешенного решения. В противоречивых ситуациях нечеткая логическая система принимает средневзвешенное решение. Например, цель слева и препятствие слева. В этом случае нечеткая логическая система обходит препятствие, слегка стремясь к цели.
- Существенно меньшее число правил по сравнению с экспертной системе. За счет возможности интерполяции и принятия средневзвешенного решения число классификаций в нечеткой логической системе на много ниже. Поэтому число правил в системе сокращается.
- Возможность оперирования с непрерывными значениями. В отличие от экспертной системы, решения в которой дискретны, в нечеткой логической системе достаточно легко можно оперировать с непрерывными значениями и получать непрерывный результат.
- Объяснимость результата нечеткого вывода. С точки зрения человеческого восприятия несложно объяснить причину принятия нечеткой логической системой того или иного решения. А если человек может объяснить решение, в том числе неправильное, то он может и предпринять меры по его предотвращению.

Однако следует отметить и недостатки, по крайне мере в классической реализации нечеткой логической системы:

- Невозможность организации памяти внутри нечеткого логического регулятора. Для работы с памятью приходится устанавливать входы и выходы, замкнутые на память.
- Невозможность проведения предварительных математических вычислений. Нечеткой логической системе следует подавать на вход уже посчитанные величины. Например, зная азимут ориентации робота и азимут цели, нечеткая логическая система

не способна самостоятельно найти разность между этими величинами, чтобы повернуть на цель. На вход в данном случае нужно подавать разность этих величин, посчитанную внешней системой.

- Невозможность организации процесса самообучения (в классической реализации). Несмотря на то, что существуют работы, в которых приводятся попытки организовать в нечеткой логической системе механизмы автоматической генерации и модификации функций принадлежности, в классической реализации нечетких логических систем таких механизмов нет.

5.4.3. Применение нечетких логических систем в составе системы управления интеллектуальных мобильных роботов

Нечеткие логические системы могут применяться там, необходимо применять интеллектуальные системы, оперирующие с непрерывными сигналами.

Нечеткая логическая система может быть применена на приводном (нижнем) уровне системы управления при разработке интеллектуальных регуляторов. На вход нечетких логических систем в этом случае следует подавать сигнал ошибки, ее интегральную и дифференциальную составляющую и, возможно, еще какой-нибудь служебный сигнал. Роль коэффициентов ПИД-регулятора может выполнять нечеткая логическая система, обеспечивающая инвариантность к инерции объекта или обеспечивающая больший диапазон рабочих скоростей объекта, когда это позволяет нагрузка. Однако применение нечеткой логической системы требует от микропроцессоров приводного уровня большей производительности и аппаратных ресурсов, чем классический ПИД-регулятор.

С большей эффективность нечеткая логическая система может быть применена на тактическом уровне системы управления (система управления движением). На тактическом уровне системе управления приходится оперировать с такими непрерывными сигналами, как пеленг и расстояние до цели, опасности направлений движения, а также находить компромисс между процессом движения к цели и процессом обхода препятствий. В большинстве случаев отдельные решения по этим двум процессам противоречат друг другу. С помощью нечетной логической системы можно достаточно легко «взвесить» данные противоречивые решения малым числом продукционных правил (в классической реализации 6 правил). Процесс «взвешивания» можно обеспечить путем изменения ширины выходных функций принадлежности термов, обеспечивающих поворот робота к целевой точке, по отношению к ширине

выходных функций принадлежности термов, обеспечивающих отклонение робота от препятствий.

Подобная система была неоднократно апробирована на различных моделях мобильных роботов и их экспериментальных образцах и показала свою эффективность. Следует отметить, что также показали свою эффективность боты врагов для 3D-игры, передвигающиеся по пересеченной местности, алгоритм работы которых был основан на нечеткой логической системе.

Хорошие результаты показывает реализация двухуровневой системы управления движением. На более высоком уровне находится нечеткая логическая система, прокладывающая путь на глобальной карте местности, анализируя глобальные опасности направлений (например, слева в 100 метрах от робота лес, справа – болото), а на более низком уровне находится нечеткая логическая система, планирующая маршрут с учетом локальных препятствий на пути (например, отдельных деревьев). Целевая точка маршрута назначается верхней системе, которая формирует для нижней системы пеленг локальной цели и расстояние до целевой точки.

Подобная система управления была успешно опробована на модели летательного аппарата, двигающегося по пересеченной местности на высотах до 10 м. Причем важно, что карта глобальной местности при этом обновлялась по мере ее пополнения разведывательными данными, и планирование маршрута производилось с учетом этих новых данных.

На стратегическом уровне системы управления (уровень управления поведением) применять нечеткую логическую систему нецелесообразно, т.к. на стратегическом уровне требуется оперировать в основном с логическими и дискретными сигналами, а принимаемое решение должно быть четким. Нельзя, например, по принципу работы нечеткой логической системы придти в точку *A* и немного в точку *B*, стратегический уровень должен четко определиться с принимаемым решением.

5.4.4. Рекомендации по программированию нечеткой логической системы

При программировании нечеткой логической системы удобно использовать редактор функций принадлежности и продукционных правил.

В данном редакторе весь диапазон входных и выходных переменных разбить на 256 дискрет. По каждому входу реализовать список функций принадлежности.

Сами функции принадлежности удобно реализовать в виде массивов из 256 элементов. В этих элементах будет храниться значение, соответствующее конкретному входному значению, приведенному в интервал от 0 до 255:

```
typedef struct
{
     char name[64];          // имя терма
     char probability;       // достоверость, выставляться при фаззификации
     unsigned char values[256]; // массив значений
} FUZZY_FUNC;
```

Значение элемента массива будет изменяться от 0 до 255. Значение 0 будет соответствовать достоверности 0, а значение 255 будет соответствовать достоверности 1.

В данном виде над функциями принадлежности легко можно производить операции обрезания и объединения, используемые в нечетком логическом выводе.

Сами продукционные правила можно хранить в текстовом виде. Они достаточно просты для реализации их программного парсера.

На каждом такте расчета нечеткой логической системы, программист должен произвести фаззификацию и в каждую входной функцию принадлежности записать ее достоверность (поле *.probability*).

Затем начать выполнять продукционные правила. Парсер правил должен разбить правило на левую и правую части. Затем начать разбор и вычисление значения выражения, стоящего между ключевым словом «ЕСЛИ» и «ТО», определить его достоверность, а затем в выходную результирующую функцию принадлежности добавить значение активированной правой частью функции принадлежности, обрезанной вычисленной достоверностью левой части.

На последнем этапе программист должен произвести дефаззификацию, т.е. найти центр масс результирующих фигур по выходным переменным. В силу дискретности представления результирующих функций принадлежности, интеграл можно заменить суммой:

$$y_{\text{вых},j} = \frac{\sum_{i=0}^{255} f_{j,i} \cdot i}{\sum_{i=0}^{255} f_{j,i}}$$

Где: $f_{j,i}$ – элементы массива значений результирующей j-ой функции принадлежности (поле *.values*[i]); $y_{\text{вых},j}$ –выходное числовой значение j-ого выхода в диапазоне от 0 до 255.

По окончании, значение $y_{\text{вых},j}$ следует перевести из диапазона от 0 до 255 в диапазон от $y_{\min,j}$ до $y_{\max,j}$, где: $y_{\min,j}$, $y_{\max,j}$ – диапазон определения данной j-ой выходной переменной.

5.4.5. Реализация нечетких логических систем в Dyn-Soft RobSim 5

Для реализации нечетких логических систем в Dyn-Soft RobSim 5 в редакторе структурных схем программного обеспечения бортовой ЭВМ имеется специальных блок «Нечеткая логическая система» (Рис. 106).

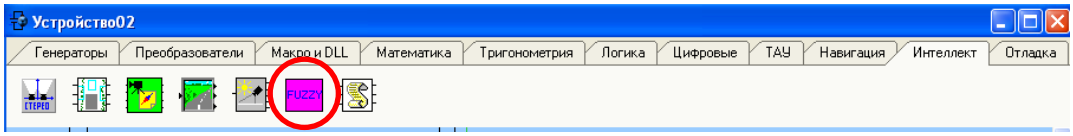

Рис. 106 Расположение блока «Нечеткая логическая система» на палитре блоков редактора структуры программного обеспечения в Dyn-Soft RobSim 5

При установке данного блока в структуру программного обеспечения изначально формируется блок без входов и выходов. Входы и выходы блока добавляются в его свойствах. Для этого в редакторе настроек нечеткой логической системы следует переключиться на закладку «Функции принадлежности» и с помощью всплывающего меню левого списка добавить требуемые входы и выходы. (Рис. 107).

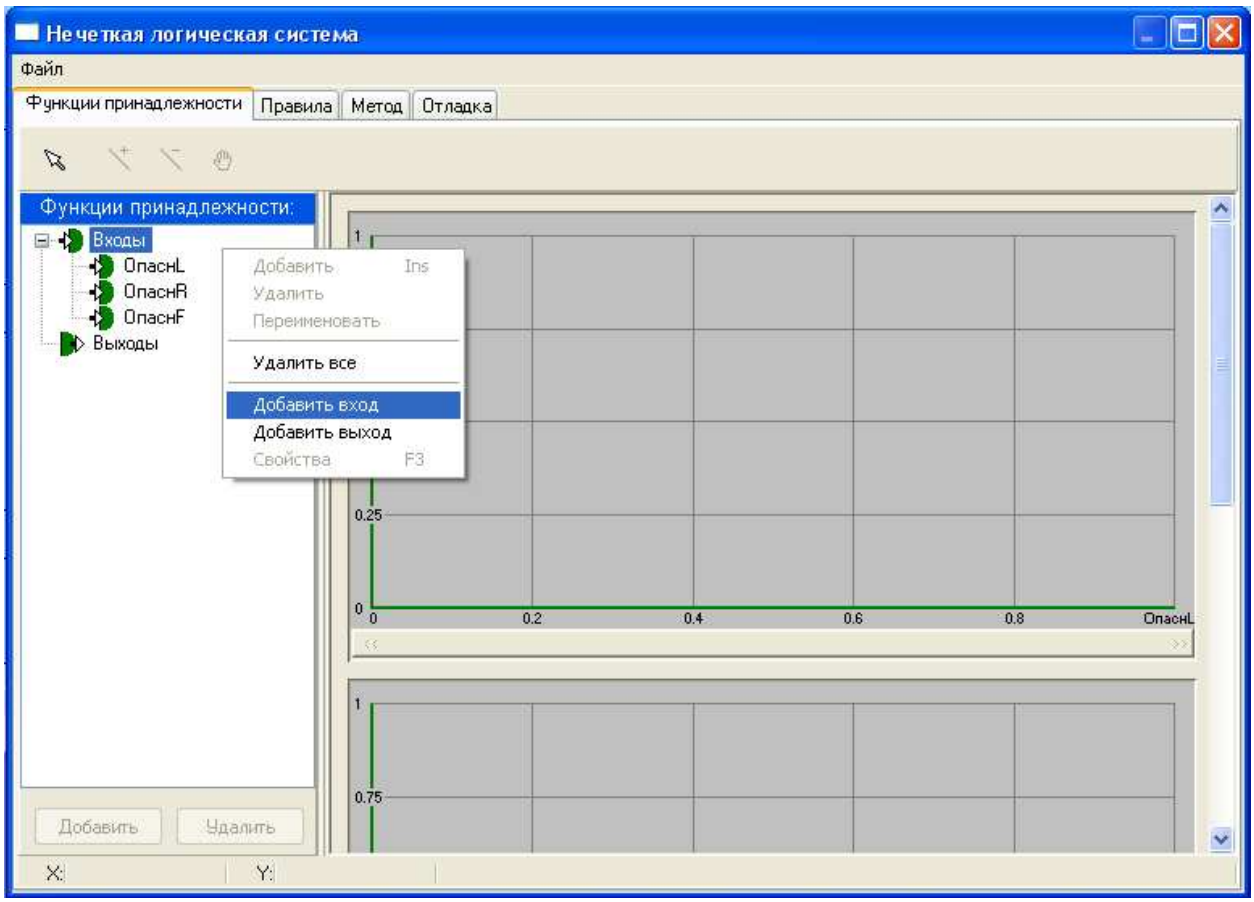

Рис. 107 Пример добавления входов и выходов в редакторе нечетной логической системы

Все входы и выходы нечеткой логической системы будут иметь тип данных float.

При добавлении входов и выходов пользователь должен задать диапазон изменения сигнала по каждому входу и выходу, путем выбора пункта «Свойства» всплывающего меню списка «Функции принадлежности» (Рис. 108).

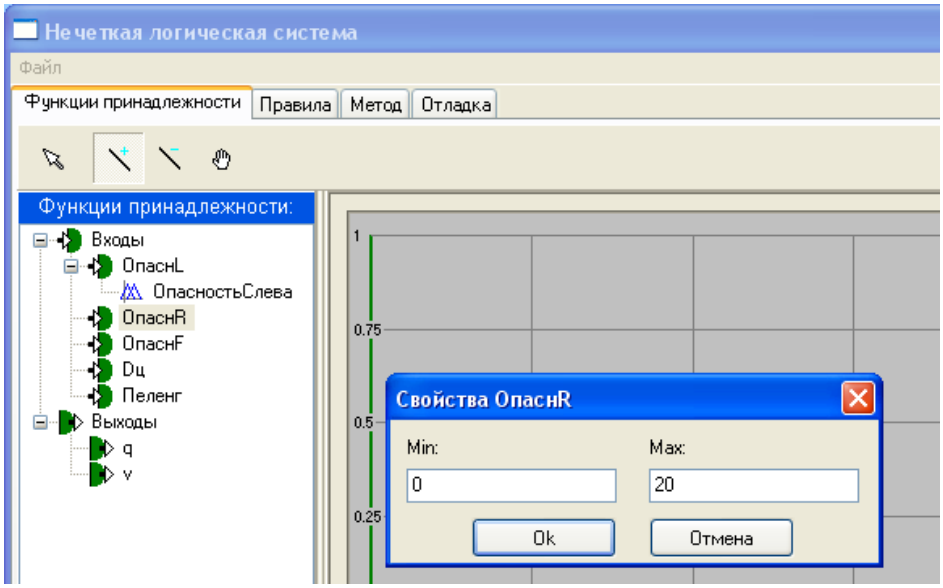

Рис. 108 Пример формирования диапазона изменения сигнала по входам и выходам блока «Нечеткая логическая система»

Затем по каждому входу и выходу пользователь должен определить термы и их функции принадлежности. Для этого следует на закладке выбрать соответствующих вход или выход, нажать кнопку «Добавить» или выбрать соответствующий пункт во вплывающем меню (Рис. 109).

Обязательно функцию принадлежности следует переименовать в название терма. Название должно быть написано в одно слово!

При активизации терма в левой списке активизируется графический редактор, позволяющий редактировать график соответствующей функции принадлежности.

Следует отметить, что график функции принадлежности представляет собой массив из 256 значений. По умолчанию значение всех дискет равно нулю. Редактор функции принадлежности имеет два основных инструмента: «Инструмент рисования» и «Инструмент стирания» (Рис. 109). «Инструмент рисования» рисует прямую; все значения дискрет ниже данной прямой принимают значения точек на прямой. «Инструмент стирания» наоборот: рисует прямую; все значения дискрет выше данной прямой принимают значения точек на прямой. Другими словами, «инструмент рисования» добавляет к графику функции

принадлежности значения, а «Инструмент стирания» обрезает эти значения.

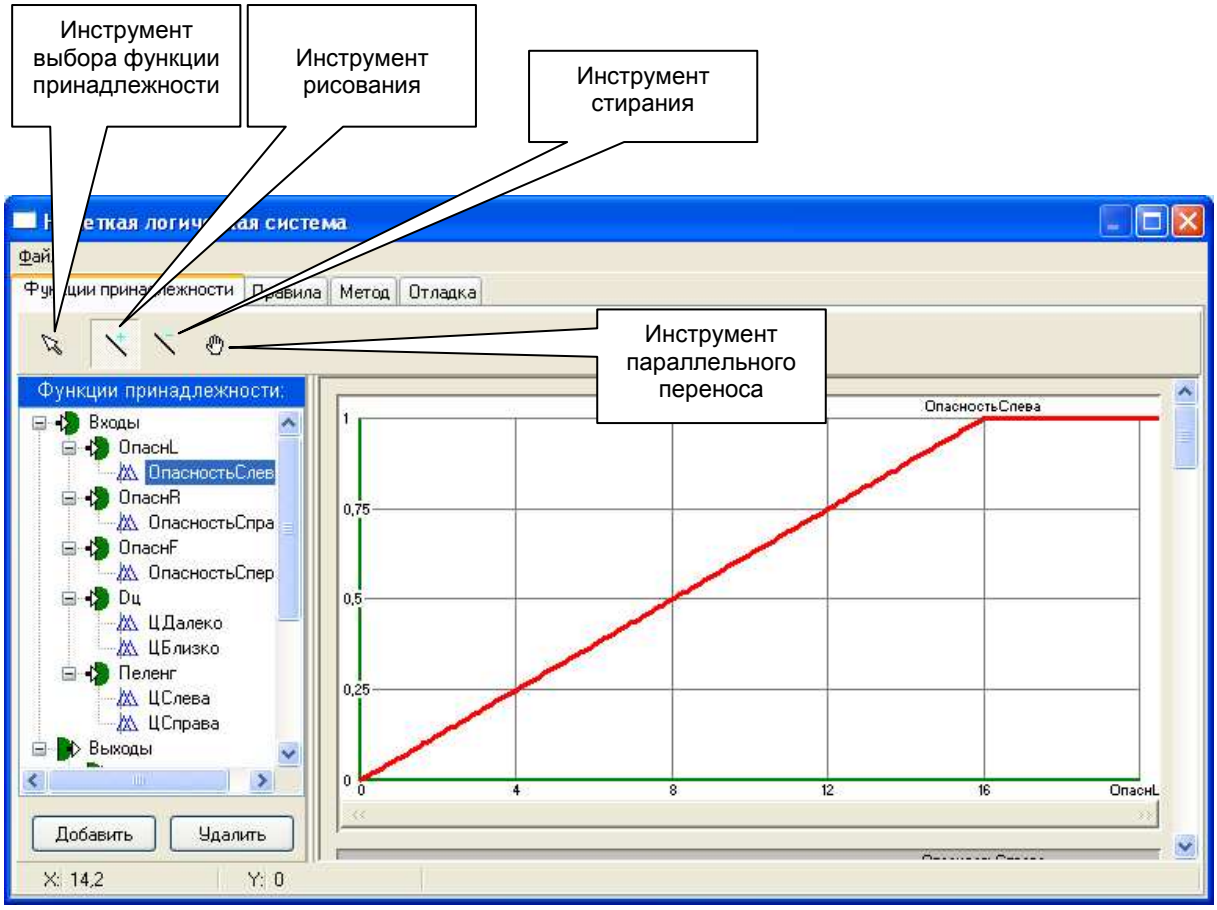

Рис. 109 Иллюстрация процесс добавления и редактирования функции принадлежности в настройках нечеткой логической системы в Dyn-Soft RobSim 5

В результате пользователь может создать график функции принадлежности любой формы.

Для удобства редактирования в состав редактора функций принадлежности всходит инструмент «Параллельный перенос». Данный инструмент позволяет перемещать график функции принадлежности по горизонтали.

Созданный график функции принадлежности при помощи пункта всплывающего меню «Копировать» можно скопировать в буфер обмена и с помощью пункта меню «Вставить» – вставить в качестве графика для другой функции принадлежности.

Также график функции принадлежности с помощью пункта меню «Отразить зеркально» можно зеркально отразить относительно вертикальной прямой. Данная функция наиболее востребована в сочетании с функцией копирования функции принадлежности.

После создания функции принадлежности в нечеткой логической системе следует задать базу продукционных правил. Для этого следует переключиться на закладку «Правила» (Рис. 110).

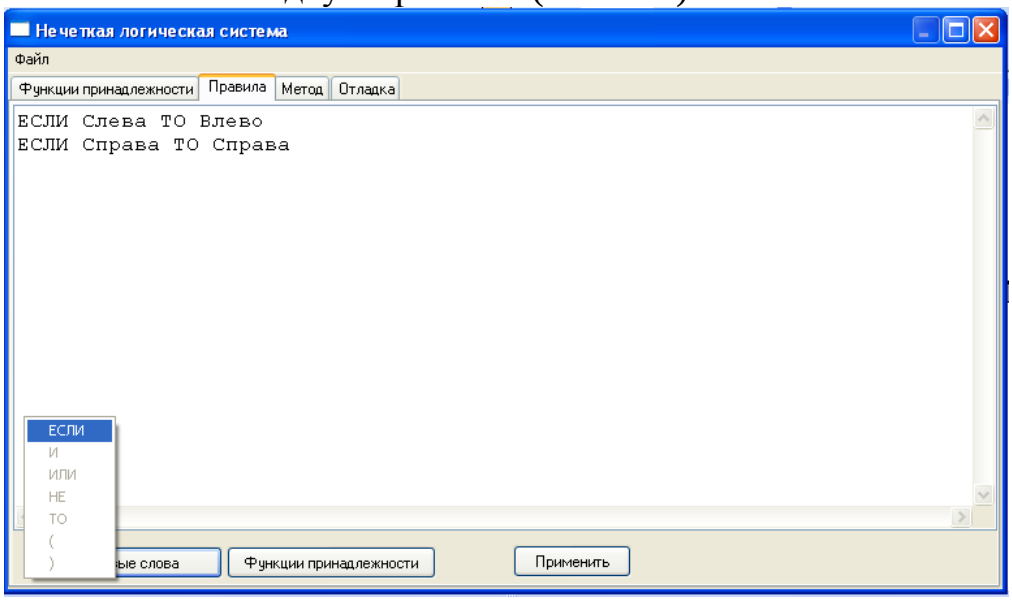

Рис. 110 Внешний вид редактора продукционных правил в редакторе настройки нечеткой логической системы в Dyn-Soft RobSim 5

С помощью редактора продукционных правил необходимо записать продукционные правила в формате:

```
ЕСЛИ <условие1> ТО <следствие1>
ЕСЛИ <условие2> ТО <следствие2>
ЕСЛИ <условие3> ТО <следствие3>
…
ЕСЛИ <условиеN> ТО <следствиеN>
```

Важное требование: по одному правилу на строку.

Здесь: <условие*i*> – перечень входных термов, активизирующих данное правило, объединенных через оператор «И», «ИЛИ» или «НЕ». <следствие*i*> – перечень выходных термов, активизирующихся данным правилом, объединенных через оператор «И».

Для помощи написания правил внизу редактора имеются кнопки с подсказками. При нажатии на кнопку появляется всплывающее меню. Доступность пунктов меню, в которых вписано ключевое слово и название терма, определяет возможность вставки данного текста в основной текст правила в данном месте.

После записи всех правил рекомендуется нажать кнопку «Применить». При этом произойдет компиляция правил и проверка на ошибки. К сожалению, закрыть редактор нечеткой логической системы невозможно, пока в тексте правил содержатся ошибки или недописанные правила.

Метод нечетного вывода «MAX-MIN» или «MAX-DOT» выбирается на закладке «Метод».

Для проверки работоспособности нечеткой логической системы используется закладка «Отладка» (Рис. 111).

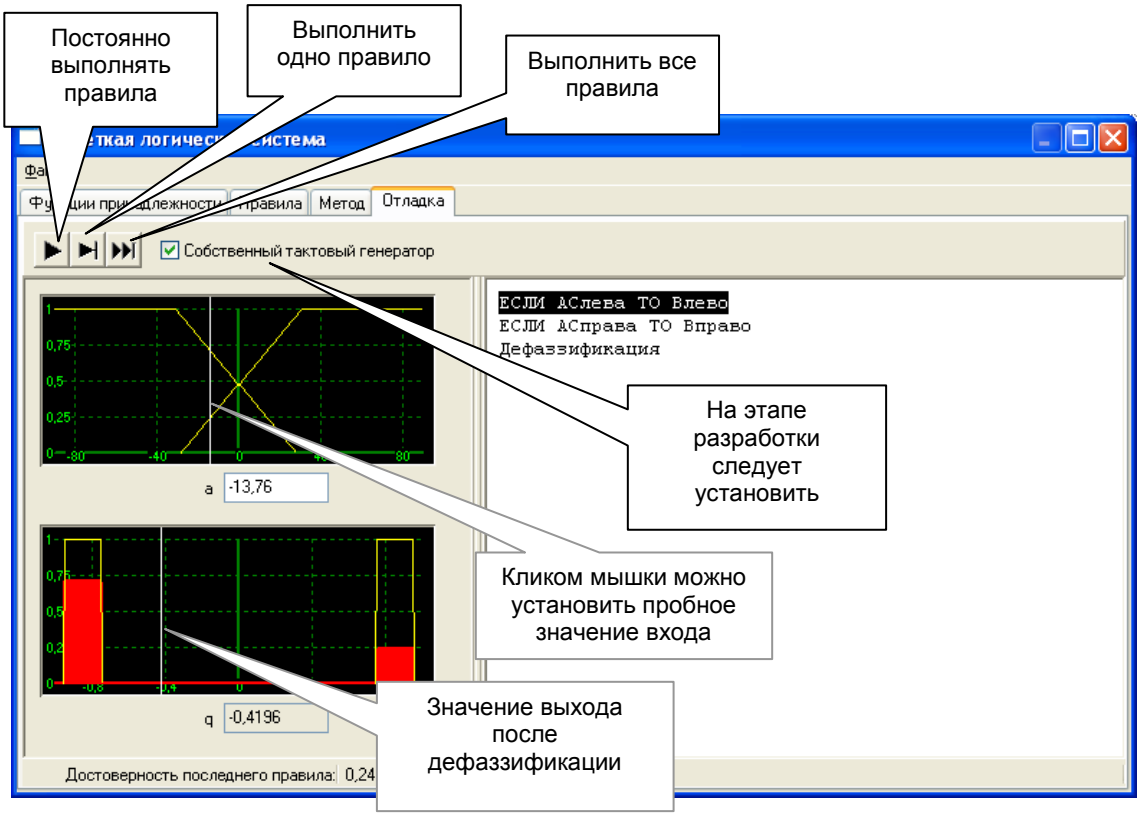

Рис. 111 Внешний вид редактора нечеткой логической системы в режиме отладки

Для отладки базы знаний правил следует обязательно установить галочку «Собственный тактовый генератор». Данная галочка определяет необходимость запуска нечеткой логической системы не средствами среды моделирования, а с помощью встроенного механизма запуска.

Путем перемещения белой вертикальной линии по графикам входных функций принадлежности пользователь может сформировать на входах проверочные входные воздействия на нечеткую логическую систему. Затем путем выполнения одного или нескольких правил можно проследить по шагам процесс нечеткого логического вывода, наблюдая за построением выходных функций принадлежности (выходные модифицированные функции принадлежности рисуются красным). После выполнения дефаззификации формируется выходное значение по каждому выходу нечеткой логической системы. Белая вертикальная линия на графике функций принадлежности по выходным переменным обозначает выходное значение (положение центра масс результирующей фигуры).

В программном 3D-симуляторе Dyn-Soft RobSim 5 во время испытания робота путем нажатия клавиши «F» можно также вызвать данный редактор для отладки работы нечеткой логической системы в условиях виртуальной сцены. При этом выставление галочки

«Собственный тактовый генератор» не требуется, а входное значение формируется системой управления робота.

Следует отметить, что данный редактор нечеткой логической системы применялся во многих других проектах, поэтому некоторые элементы интерфейса данного редактора унифицированы или использованы под нужды других проектор.

5.4.6. Пример реализации системы управления движением мобильного робота в среде с препятствиями на базе нечеткой логической системы в Dyn-Soft RobSim 5

Как уже отмечалось, нечеткую логическую систему удобно использовать для создания системы управления движением мобильного робота в среде с препятствиями. В данной главе приводится пример реализации такой системы на примере робота в Dyn-Soft RobSim 5.

Структурная схема подключения нечеткой логической системы, обеспечивающей автономное движение мобильного робота к целевой точке в среде с препятствиями, в структуру программного обеспечения бортовой ЭВМ приведена на Рис. 112.

На данной схеме реализованы механизмы формирования локальной карты местности (см. главу 3.2.4), анализаторы опасности которой формируют на выходе сигналы опасности препятствий слева, справа и прямо по курсу. Данные сигналы непосредственно поступают на нечеткую логическую систему на входы «ОпаснL», «ОпаснR» и «ОпаснF» соответственно.

Также на схеме имеется система навигации (см. главу 4), обеспечивающая на выходе формирование координат робота $X_р$, $Y_р$ и азимутального угла его ориентации $\alpha_р$. Технология и подробности реализации системы навигации робота в данном примере не рассматриваются.

В данном примере от пульта оператора приходит сигнал *mode*, который может принимать следующие значения:

- 0 – ручное управление роботом;
- 1 – режим движения к целевой точке 1;
- 2 – режим движения к целевой точке 2.

Способ формирования данного сигнала был рассмотрен в примере с экспертной системой в главе 5.2.4.

В зависимости от сигнала *mode* мультиплексоры выбирают в качестве целевой точки $X_ц$ и $Y_ц$ константы $X_{ц1}$ и $Y_{ц1}$ (если *mode*=1), или $X_{ц2}$ и $Y_{ц2}$ (если *mode* = 2).

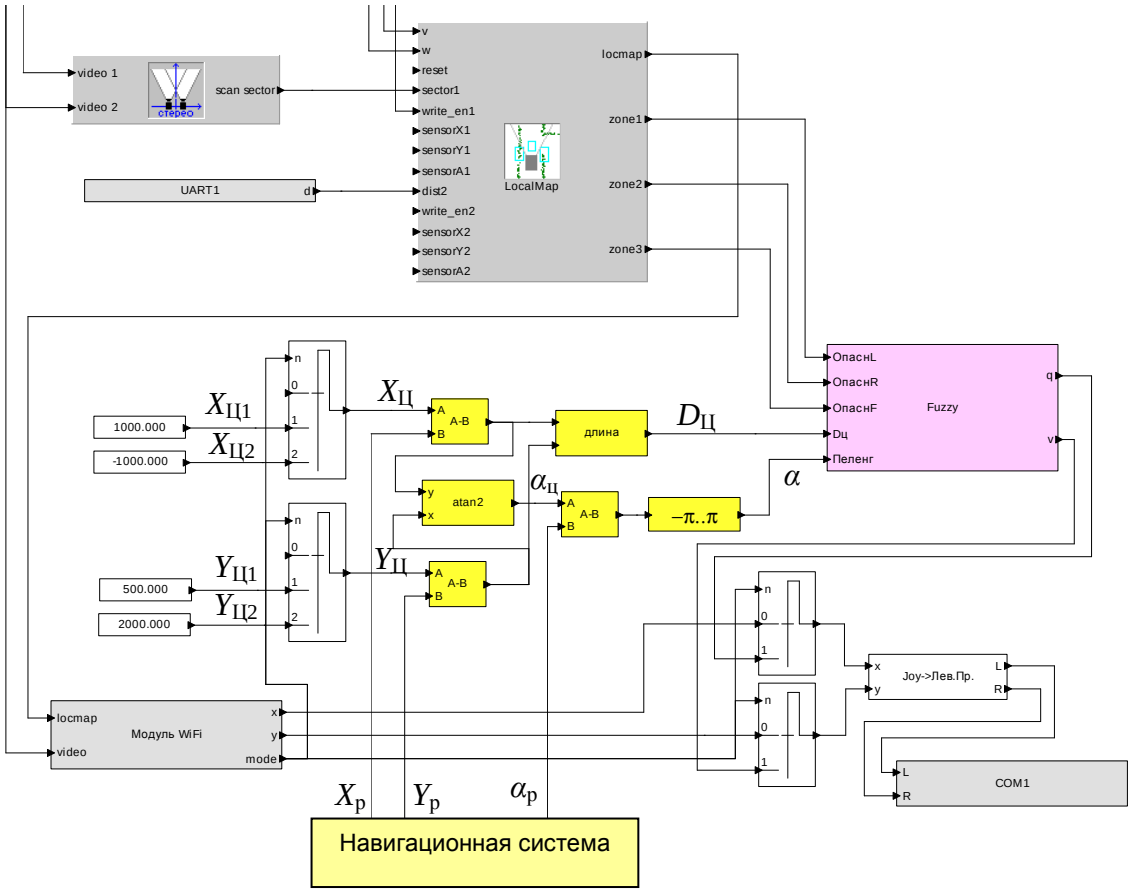

Рис. 112 Структурная схема подключения нечеткой логической системы, обеспечивающей движение в среде с препятствиями, в структуру программного обеспечения бортовой ЭВМ

По аналогии со схемой, приведенной в главе 5.2.4, координаты целевой точки, координаты робота и его ориентация пересчитываются в расстояние до целевой точки $D_{ц}$, и ее пеленг α, которые поступают на вход нечеткой логической системы.

Нечеткая логическая система формирует сигнал задания на скорость поворота q и сигнал задания на линейную скорость движения робота v.

Выходные сигналы нечеткой логической системы через выходные мультиплексоры поступают на блок «Пересчет координат джойстика в сигнал левого и правого привода», на выходе которого формируется уставка скорости для левого и правого борта. Связь с контроллером управления двигателями осуществляется через порт COM1.

Также схема предусматривает контур ручного управления. В режиме *mode*=0 выходные мультиплексоры переключены на прием сигналов ручного управления, а в режимах 1 и 2 – переключены на выход нечеткой логической системы.

База знаний нечеткой логической системы состоит из продукционных правил и функций принадлежности, приведенных на Рис. 113 и Рис. 114.

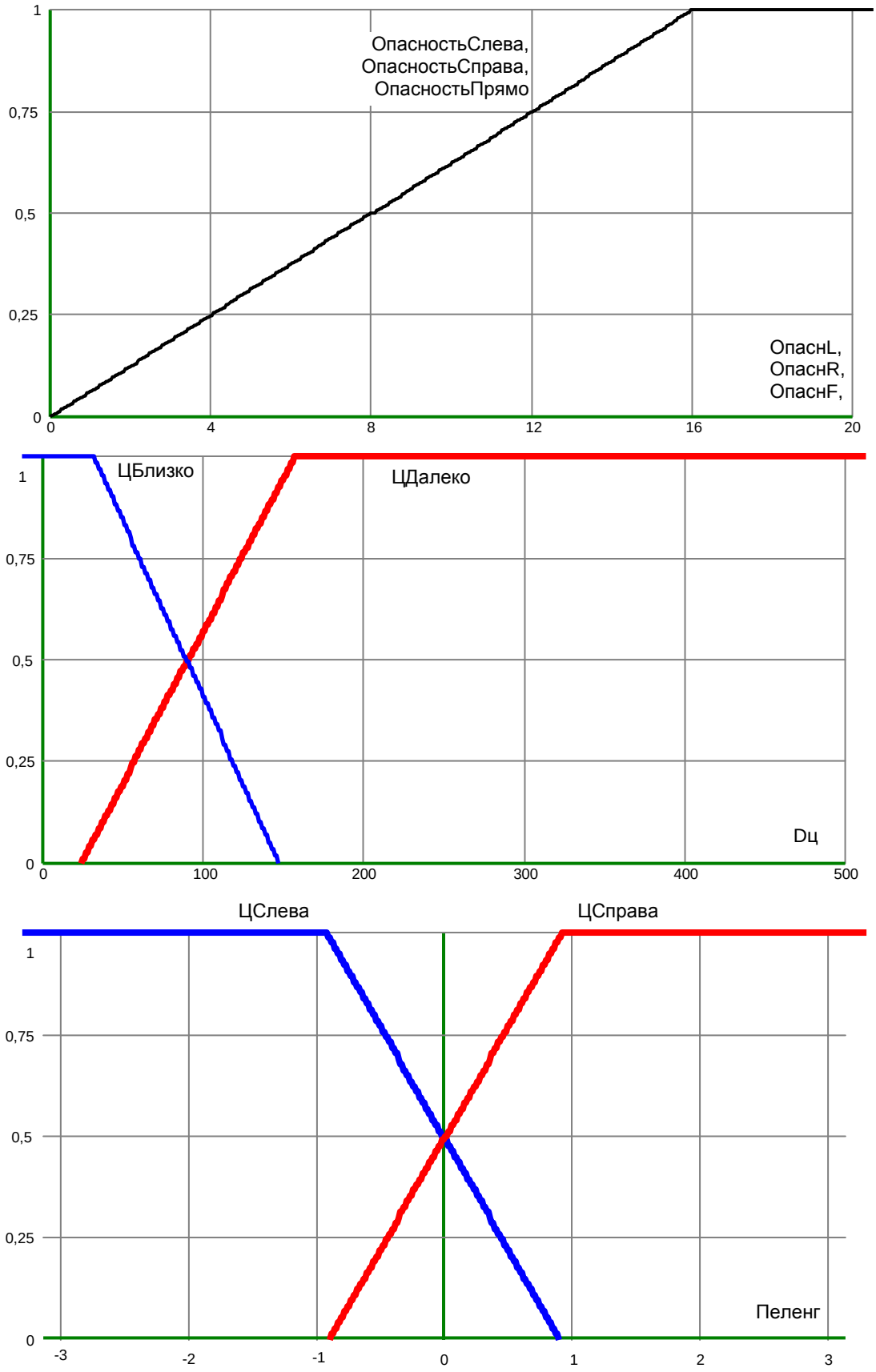

Рис. 113 Функции принадлежности по входным переменным «ОпаснL», «ОпаснR», «ОпаснF», «Dц» и «Пеленг»

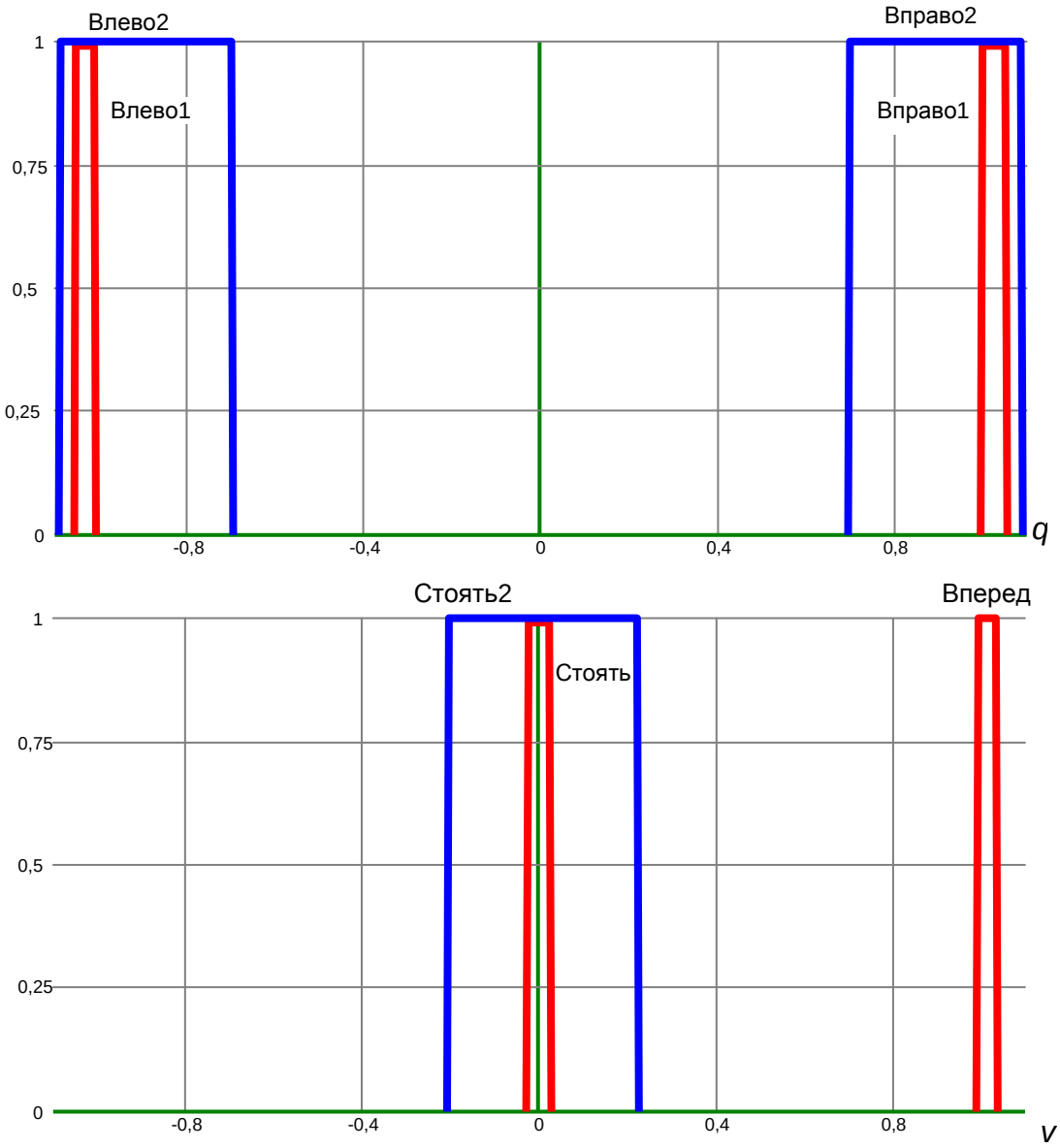

Рис. 114 Функции принадлежности по выходным переменным q и v

Функции принадлежности по входным переменным «ОпаснL», «ОпаснR» и «ОпаснF» похожи, поэтому изображены на одном графике (хотя их следует формировать по разным входам).

Функции «ЦСлева» и «ЦСправа» симметричны относительно нуля. Их симметрия крайне важна при реализации нечетной логической системы. Нарушение симметрии влечет наличие постоянного поворота робота вместо движения прямо. Функции принадлежности «Влево1» и «Вправо2», равно как функции принадлежности «Влево2» и «Вправо2» также симметричны.

Выходные функции принадлежности «Стоять» и «Стоять2» должны быть расположены так, чтобы их центр масс находится в нуле по оси скорости *v*. В противном случае полная остановка робота у целевой точки будет невозможна.

Продукционные правила нечетной логической системы следующие:

1. ЕСЛИ ЦСлева ТО Влево1
2. ЕСЛИ ЦСправа ТО Вправо1
3. ЕСЛИ ЦДалеко ТО Вперед
4. ЕСЛИ ЦБлизко ТО Стоять
5. ЕСЛИ ОпасностьСлева ТО Вправо2
6. ЕСЛИ ОпасностьСправа ТО Влево2
7. ЕСЛИ ОпасностьСперед ТО Стоять2

Согласно правилам 1 и 2, при активизации термов «ЦСлева» и «ЦСправа» активизируются термы «Влево1» и «Вправо1». Функции принадлежности входных термов специально были сформированы пересекающимися. Т.е. если переменная «Пеленг» равна 0, то достоверность правила «ЦСлева» и «ЦСправа» равна 0.5, и выходные симметричные функции принадлежности «Влево1» и «Вправо1» также активизируются по уровню 0.5. При этом центр масс выходной фигуры, образованной этими функциями принадлежности находится в точке q=0. Однако стоит пеленгу цели стать чуть больше нуля или чуть меньше, то выходные функции принадлежности будут активизироваться разными уровнями, поэтому центр масс результирующей фигуры будет смещаться вправо или влево от нуля, т.е. роботом будет производиться поворот в направлении целевой точки.

При угле пеленга 0.9 радианы происходит насыщение входной функции принадлежности «ЦСправа», а достоверность функции принадлежности «ЦСлева» становится равной нулю. При этом будет активизирована только выходная функция принадлежности «Вправо1», и центр масс будет находиться в точке q=0.9. При этом будет достигаться максимальная скорость поворота робота. Аналогичная ситуация, только в другую сторону, будет наблюдаться при угле пеленга -0.9 радианы.

Согласно правилам 3 и 4, если цель далеко, то активизируется функция принадлежности «Вперед», а если цель близко, то активизируется выходная функция принадлежности «Стоять». При этом при расстоянии до целевой точки от 20 до 130 см производится плавный переход из состояния «Вперед» в состояние «Стоять». Т.е. при подходе к цели робот будет плавно сбрасывать скорость до полной остановки.

Согласно правилами 5 и 6 производится обход препятствий. При активизации терма «ОпаснСлева» активизируется выходной терм «Вправо2», а при терма «ОпаснСправа» – «Влево2». Следует отметить, что удельный вес функций принадлежности «Вправо2» и «Влево2» намного больше, чем у функций принадлежности «Вправо1» и «Влево1». Т.о., даже небольшая активизация функций принадлежности «Вправо2» и «Влево2» перетягивает центр масс выходной фигуры на себя. За счет этого робот имеет больший приоритет поворота при обходе препятствий, чем для

движения к целевой точке. Однако, к данному роботу уместно употребить выражение «обходит препятствие, слегка стремясь к цели».

Следует заметить, что чем ширине ширина функций принадлежности «Вправо2» и «Влево2» по сравнению с шириной функций «Влево1» и «Вправо1», тем более робким получается робот. И наоборот, чем меньше ширина функций принадлежности «Вправо2» и «Влево2», тем более смелым получается робот. Важно соблюдать изначальную симметрию функций.

Согласно правилу 7, если перед роботом возникает препятствие, то робот останавливается. При этом активизируется терм «Стоять2», удельный вес которого доминирует над весом функции принадлежности «Вперед», поэтому даже при небольшой активизации терма «Стоять2» производится резкий сброс скорости.

При отладке системы управления разработчику рекомендуется сначала провести испытания работы системы на открытой местности, реализовав только правила 1-4. При этом важно добиться движения к целевой точке с приемлемой скоростью поворота, без перерегулирования и с удовлетворительной точностью прихода к цели. Для обеспечения скорости поворота рекомендуется варьировать входными функциями принадлежности «ЦСлева» и «ЦСправа», а для достижения удовлетворительной точности – «ЦДалеко» и «ЦБлизко».

Если движение к цели на открытой местности роботом производится адекватно, то стоит реализовать правила 5-7, и производить испытания в среде с препятствиями. При этом, варьируя шириной функций принадлежности «Влево2» и «Вправо2» (соблюдая их симметрию), следует добиться приемлемого радиуса обхода препятствий на пути к целевой точке.

В целом следует отметить, что данная базовая настойка из 7 правил может при необходимости дополняться другими правилами, обеспечивающие роботу какие-либо специальные аспекты поведения. Например, правила запрета частого поворота или выбора направления выхода из тупиков.

5.5.Нейронные сети

5.5.1. Описание технологии нейронных сетей

Технология нейронных сетей – это попытка создания интеллектуальной системы по аналогии с нейронной сетью живых организмов.

При изучении нервных клеток (нейронов) живых организмов ученые выяснили, что функционирование отдельного нейрона достаточно простое: на вход нейрону поступают раздражители (по сути, входные сигналы),

нейрон их складывает, пропускает через некоторую специфичную только для данного типа нейронов нелинейную функцию и формирует выходной сигнал. В живых нейронах уровень сигнала определяется параметрами частотно-импульсной модуляции (ЧИМ), в искусственных нейронах уровень сигнала определяется цифровым значением входа.

Сам по себе нейрон не обладает каким-либо интеллектом. Интеллект нейронной сети (как живой, так и искусственной) заключается в связях нейронов друг с другом (весовых коэффициентах).

Структура искусственного нейрона представлена на Рис. 115 (а). Она представляет собой весовой сумматор с нелинейным элементом на выходе. Количество входов у нейрона может быть различное, в основном зависит от количества нейронов в предыдущем слое (см. далее). Каждое входной сигнал умножается на свой весовой коэффициент и складывается с результатами таких же умножений для других входов. Полученная сумма пропускается через нелинейный элемент, выход которого является выходным значением нейрона.

Нелинейная функция нейрона может меняться в зависимости от назначения нейрона. Чаще используется гладкая функция, описываемая функцией:

$$z(x) = -\frac{\operatorname{arctg}\left(\left(x - 0.5\right) \cdot K\right)}{\pi}$$

Здесь: K – коэффициент жесткости, чем он больше, тем менее плавный переход из состояния 0 в состояние 1. Рекомендуется использовать K=10.

Используют и другие нелинейные функции нейронов, например, релейную характеристику (для реализации четкого переключения выходных состояний), а также прочие нелинейные функции.

Из отдельных нейронов строится нейронная сеть (Рис. 115, б). В ней нейроны обычно располагаются слоями. Число нейронов в каждом слое может быть различным. Каждый нейрон следующего слоя связан со всеми нейронами предыдущего слоя. У каждой такой связи имеется свой вес.

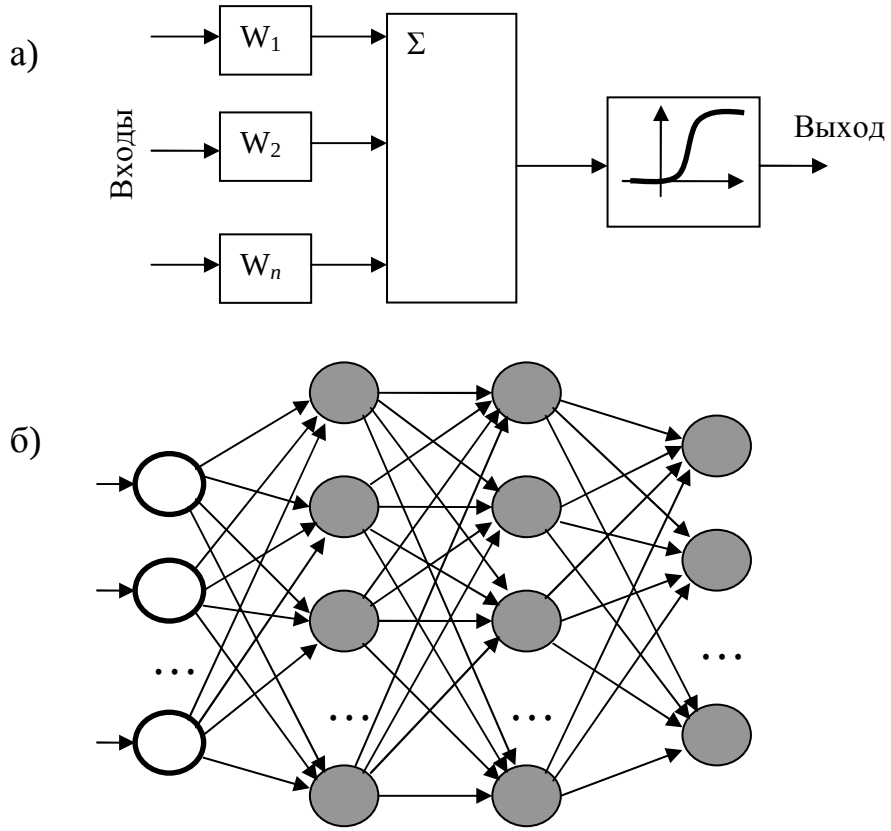

Рис. 115 Структура одного нейрона (а); структура нейронной сети (б).

На входе нейронной сети располагаются специальные нейроны, представляющие собой рецепторы. В искусственной нейронной сети входные нейроны – это просто входы интеллектуальной системы.

Настройка нейронной сети осуществляется подбором весовых коэффициентов каждого нейрона. В сложной нейронной сети путем подбором весовых коэффициентов можно реализовать практически любую многовходовую нелинейную функцию по нескольким выходам.

Вручную подобрать веса нейронов так, чтобы нейронная сеть формировала на выходе требуемый результат, практически невозможно. Поэтому подбор весовых коэффициентов нейронов осуществляется только путем *самообучения*.

Существует два способа обучения сети:

- Обучение на примерах;
- Обучение с критерием качества.

В случае обучения на примерах для нейронной сети подготавливают больше количество контрольных примеров, в которых для определенных ключевых значений входа имеется требуемое значение выхода. Для всех этих примеров запускается процесс самообучения, в результате которого подбираются коэффициенты нейронной сети таким образом, чтобы на выходе для всех входных воздействий из примеров формировался

требуемый результат с минимальной ошибкой. После обучения нейронная сеть интерполирует выходные значения, тем самым становится способной принимать решения даже при тех значениях входных воздействий, которым ее не обучали.

В ряде случаев примеры правильной работы системы создать невозможно. Например, необходимо создать робота, который с минимальной затратой энергии, обходя препятствия, добрался бы до целевой точки за минимальное время. Следуя данной логике, без знания объекта управления даже человеку сложно составить обучающие примеры.

В этом случае применяют обучение с критерием качества. Для этого подготавливают скалярную величину, являющуюся оценкой (критерием) качества работы системы. Чем меньше значение этого критерия, тем лучше работает система. Разработчик критерия может не знать, как достигнуть требуемого результата, но он может обучить нейронную сеть следовать данному критерию.

После подготовки критерия производится процесс обучения нейронной сети. При этом проводят множество различных экспериментов (обычно сначала на модели системы) и пытаются таким образом настроить весовые коэффициенты нейронов, чтобы нейронная сеть обеспечивала минимальное значение критерия.

При детальном рассмотрении, метод обучения на примерах является частным случаем обучения с критерием качества, только в качестве критерия используется разница между эталонным и реальным значением выходов.

Существует несколько способов самообучения сети, но в основном все они являются разновидностью математического метода градиентного спуска. В том числе известный метод обратного распространения ошибки.

При обучении определяется производная по каждому весу, из них собирается вектор (градиент). Градиент нормируется и умножается на шаг перенастройки весов. Затем происходит шаг перенастройки весовых коэффициентов в обратном направлении градиента. Если после перенастройки весов значение критерия уменьшилось, следовательно, перенастройка весов производится в правильном направлении, в противном случае шаг перенастройки уменьшается вдвое. В конце концов, нейронная сеть достигает глобального или локального минимума, а шаг обучения становится пренебрежимо малым.

В ряде случаев нейронная сеть обучается за 1000-100 000 экспериментов в зависимости от размера сети.

Знания в нейронной сети представляют собой настройку весовых коэффициентов. Механизм логического вывода основан на расчете модели отдельных нейронов в составе всей нейронной сети. Процесс обучения реализован по средствам самообучения.

5.5.2. Достоинства и недостатки нейронных сетей

К достоинствам нейронных сетей можно отнести их следующие особенности:

- Возможность самообучения.
- Возможность оперирования и интерполяции с непрерывными величинами.
- Возможность распараллеливания алгоритмов расчета нейронной сети. В том числе возможность реализации на ПЛИС.
- Сходство и живыми организмами.

Однако следует отметить и недостатки:

- Невозможность объяснения результата. Человеческий ум не может понять и объяснить, почему нейронная сеть приняла именно такое решение при конкретном состоянии входов.
- Невозможность реализации ручного обучения (по той же причине).
- Довольно частое достижение в процессе обучения локального минимума критерия, вместо глобального.
- Продолжительность процесса обучения. В ряде случаев один эксперимент с реальным роботом может длиться минуты. Несложно посчитать, сколько времени требуется проведение хотя бы 1000 экспериментов.

5.5.3. Применение нейронных сетей в составе системы управления интеллектуальных мобильных роботов

Технологию нейронных сетей удобно применять для реализации различных уровней системы управления интеллектуального мобильного робота, особенно в тех местах, где требуется организация процесса самообучения.

На нижнем (приводном) уровне системы управления удобно применять технологию нейронных систем для реализации интеллектуальных регуляторов с возможностью их самообучения. При этом реализуется ПИД-регулятор, коэффициенты звеньев которого реализуются через нейронную сеть. Однако целесообразность такого использования определяется как вычислительными ресурсами, так и готовностью пользователя к организации долгого процесса самообучения системы.

На тактическом уровне системы управления также можно использовать технологию нейронных сетей. Однако обучение такой системы целесообразно произвести сначала на быстродействующей модели, а затем переносить в реального робота, где производить лишь небольшое доубучение. Пользователь должен быть готов к довольно

продолжительному времени офф-лайнового обучения системы с учетом реализации всего процесса моделирования движения робота к целевой точке на каждой итерации процесса обучения.

На стратегическом уровне системы управления можно реализовать нейронную сеть, но т.к. на стратегическом уровне системы управления приходится манипулировать десятками входных и выходных сигналов, время офф-лайнового обучения может быть очень длительным (измеряться днями, а то и неделями). Да и реализация обучающей выборки для стратегического уровня системы управления требует от разработчика достаточно сложных манипуляций.

5.5.4. Нейронные сети в Dyn-Soft RobSim 5

В программном комплексе Dyn-Soft RobSim 5 нет блоков, реализующих нейронные сети. Причина их отсутствия заключается в невозможности разумной реализации в этой программном комплексе механизма самообучения нейронной сети.

Предполагается, что пользователь, решивший использовать нейронную сеть, имеет ее программную реализацию со встроенным механизмом самообучения. Предполагается, что такой пользовать может самостоятельно провести офф-лайновый процесс самообучения системы на собственных высокоскоростных моделях робота, или предъявит для обучения требуемое количество примеров в том виде, в котором требует его реализация нейронной сети.

После обучения нейронной сети в Dyn-Soft RobSim 5 имеется возможность интеграции с этой нейронной сетью по средствам написания плагина на языке C++ и Dephi.

Пример реализации плагинов и инструкция по их написанию находятся в папке Plugins программного комплекса Dyn-Soft RobSim 5.

5.6. Ассоциативная память

5.6.1. Описание технологии ассоциативной памяти

Интеллектуальная технология ассоциативной памяти не является самостоятельным отдельным видом интеллектуальных систем. Она лишь является механизмом реализации той или иной другой интеллектуальной технологии.

Ассоциативная память представляет собой набор гиперповерхностей (гиперповерхность – трехмерная фигура, построенная в *n*-мерном пространстве). Размерность поверхности определяется числом входом интеллектуальной системы. Количество гиперповерхностей определяется числом выходов.

Для составления гиперповерхности необходимо наличие обученной интеллектуальной системы по технологии экспертных систем, нечеткой логики или нейронных сетей. Главное, наличие системы без памяти. На входы данной интеллектуальной системы поступают всевозможные сочетания входных воздействий, а значение выхода записывают в виде значения точки гиперповерхности.

В результате формируются большие многомерные массивы соответствия входных и выходных сигналов. Скорость принятия решения при данной реализации равна скорости доступа к ячейке памяти. При этом на принятие решения не затрачиваются вычислительные ресурсы бортовой ЭВМ, что дает возможность создания интеллектуальной системы даже на слабом микропроцессоре (например, на AVR).

Несложно заметить, что при наличии большого числа входов объем памяти для хранения гиперповерхности достаточно большой. Так, например, при наличии даже трех входов типа char, размерность гиперповерхности будет составлять 16.7 МБ! Однако существуют способы оптимизации этого объема.

Обычно интеллектуальные системы внутри производят классификацию входных воздействий. Достаточно большие диапазоны входных значений могут соответствовать одной классификации. Поэтому для оптимизации гиперповерхности можно ввести массив предварительной классификации, позволяющий уменьшить размерность гиперповерхности.

Если взять, например, в качестве исходной интеллектуальной технологии нечеткую логику, то можно заметить, что большую часть входного диапазона функции принадлежности по соответствующему входу имеют одинаковое значение. Поэтому для уменьшения объема памяти можно использовать одномерный массив начальной перекодировки. На вход массива будет поступать входное воздействие, а на выходе будет формироваться значение входа для гиперповерхности с уменьшенной размерностью (Рис. 116). На Рис. 116 (а) изображена исходная гиперповерхность. Ее размерность 256 x 256 = 65536 байт.

На Рис. 116 (б) изображена гиперповерхность после оптимизации. По каждому входу к гиперповерхности добавлен массив перекодировок. Используя данные массивы, удалось уменьшить размерность гиперповерхности до размеров 5 x 5. При этом объем затраченной памяти для хранения гиперповерхности и массивов перекодировок составляет: 5 x 5 + 256 + 256 = 537 байт.

Существуют и другие механизмы оптимизации гиперповерхности. Например, за счет уменьшения точности выходных значений можно ввести больший шаг дискретизации входных значений. В этом случае будет еще больше одинаковых решений, а стало быть, можно сократить размерность гиперповерхности.

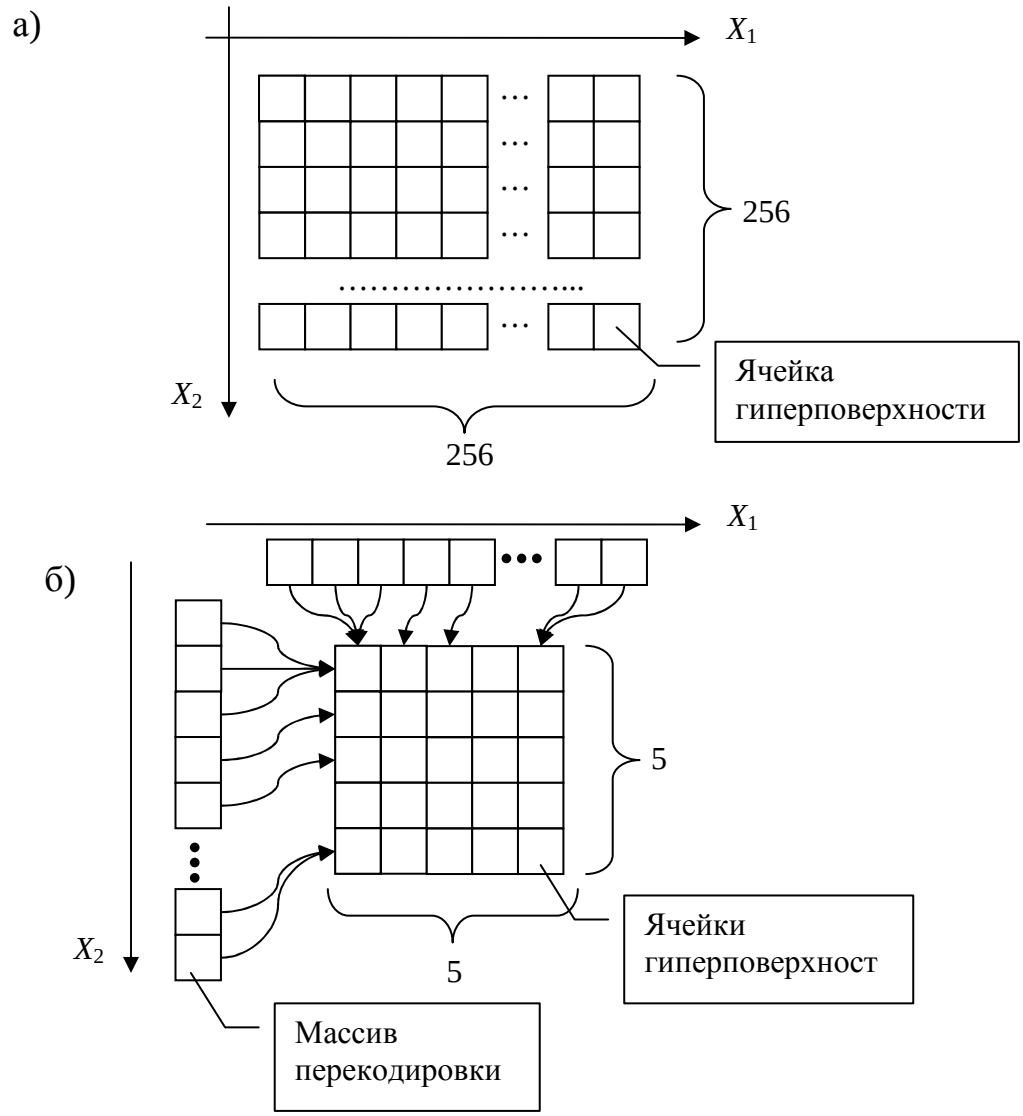

Рис. 116 Примеры гиперповерхности: а) до оптимизации; б) после оптимизации.

Кроме того, можно сокращать количество бит для хранения результата. Например, в одном байте хранить два 16-битных значения.

Также имеет смысл развязывать входы так, чтобы выходные значения зависели от малого количества входов. Так, например, если система имеет входы a, b, c и выход y_1, который зависит только от a и b, и выход y_2, который зависит только от a и c, то вместо создания гиперповерхностей для y_1 размерности a x b x c и для y_2 размерности a x b x c, можно создать гиперповерхности размерности a x b и a x c.

Если взаимная зависимость входов все же есть, то в некоторых случаях ее можно учесть каким-либо другим способом.

5.6.2. Достоинства и недостатки технологии ассоциативной памяти

К достоинствам технологии ассоциативной можно отнести:

- Высокое быстродействие (зависит лишь от скорости доступа к ячейке памяти).
- Защиты алгоритма от вскрытия (злоумышленник, даже получив доступ к данным, хранящимся в ассоциативной памяти, не может вскрыть знания, заложенные в интеллектуальную систему).

Недостатки:

- Большие объемы памяти.
- Невозможность объяснения результата.
- Большие трудозатраты организации процесса офф-лайного обучения и обеспечения оптимизации объема памяти.

5.6.3. Применение технологии ассоциативной памяти в составе системы управления интеллектуальных мобильных роботов

Технология ассоциативной памяти за счет своего высокого быстродействия удобна для реализации интеллектуальных регуляторов на нижнем (приводном) уровне системы управления, где, обычно, применяются достаточно небыстродействующие микропроцессоры. Низкое быстродействие таких микропроцессоров компенсируется высокой скорость принятия решения в технологии ассоциативной памяти без необходимости применения ресурсоемких вычислительных задач.

На тактическом уровне системы управления также можно использовать технологию ассоциативной памяти, но целесообразность ее применения имеется только в случае необходимости защиты алгоритма и знаний, заложенных в систему. Во всех остальных случаях удобнее реализовать исходную интеллектуальную технологию.

На стратегическом уровне системы управления возможно также использовать технологию ассоциативной памяти, однако только в том случае, если стратегический уровень реализован, как система без памяти. Целесообразность применения технологии ассоциативной памяти на стратегическом уровне та же, что и на тактическом уровне.

5.6.4. Использование технологии ассоциативной памяти в Dyn-Soft RobSim 5

В чистом виде технология ассоциативной памяти в программном комплексе Dyn-Soft RobSim 5 не представлена в силу неоднозначности организации процесса ее обучения. В любом случае, пользователь может использовать технологию ассоциативной памяти в качестве плагина для этого программного комплекса.

Косвенное применение технологии ассоциативной памяти в Dyn-Soft RobSim 5 все же имеется. Речь идет о реализации табличного умножения

при расчетах коэффициентов усиления пропорциональных, дифференциальных и интегральных звеньев. По сути, умножение по таблице, реализованное в перечисленных звеньях, это одновходовая ассоциативная память.

Литература:

1. Евстигнеев Д.В. Проектирование роботов и робототехнических систем в Dyn-Soft RobSim 5. Часть I.
2. Евстигнеев Д.В. Проектирование роботов и робототехнических систем в Dyn-Soft RobSim 5. Часть II.
3. Евстигнеев Д.В. Аппаратно-программное обеспечение интеллектуальных мобильных роботов. Диссертационная работа.