ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
Государственное образовательное учреждение высшего профессионального образования
«Российский государственный гуманитарный университет»

ИНСТИТУТ ЛИНГВИСТИКИ

Кафедра математики, логики и
интеллектуальных систем
в гуманитарной сфере

Волкова Татьяна Александровна

ЭЛЕМЕНТЫ ИНТЕЛЛЕКТУАЛЬНОЙ СИСТЕМЫ ТИПА ДСМ ДЛЯ УПРАВЛЕНИЯ МОБИЛЬНЫМ РОБОТОМ

Специальность № 031302 «Интеллектуальные системы в гуманитарной сфере»
Дипломная работа студентки 5-го курса очной формы обучения

Допущен(а) к защите на ГЭК

Заведующий кафедрой
д.ф.-м.н., профессор
_____Е.М. Бениаминов

«_____»_____2010 г.

Научный руководитель
к.т.н.
_____Д.А. Добрынин
(подпись)

Оценка:_____

Москва 2010

Abstract

This work describes an approach to build an adaptive robot which could be taught certain behavior in changing environment by a human teacher. For that JSM-method (a machine learning method) is used, which can generalize facts from samples and then predict unknown properties of new samples. New modification of classical JSM strategy (called JSM with excludable samples) is described. A complex containing the mobile robot, communication channel and intellectual JSM system which runs on PC was built. Also some intellectual robots' related issues are described.

Содержание

Введение

Значение интеллектуальных роботов в современном мире

На сегодняшний день интеллектуальные роботы вышли из области чисто научных разработок и становятся такими же необходимыми элементами повседневной жизни, как телевидение и сотовая связь. Такие роботы востребованы в промышленности, медицине, исследовании космоса, сфере досуга и домашнего хозяйства, военном секторе [6].

Роботов, которых можно было бы с полным правом назвать интеллектуальными, сейчас назвать сложно. Однако адаптивные роботы существуют.

Одной из базовых способностей интеллекта В.К.Финн называет способность к *адаптации* в условиях изменения жизненных ситуаций и знаний, что означает коррекцию имеющихся знаний («теории») и поведения [12].

Робот живет в изменяющейся среде. Значит, он должен быть способен адаптироваться к изменениям среды. В гибкой системе робота не должно быть абсолютных правил. Робот должен уметь переобучаться в любой момент при поступлении сигнала поощрения или наказания.

Кроме того, робот должен быть устойчивым к шумам и выбросам в данных. Это накладывает ограничения на используемые методы обучения: они должны допускать ошибки из-за несрабатывания одного из датчиков или недосмотра учителя.

Исследования в области адаптивных роботов ведутся достаточно давно. Автор имел возможность ознакомиться с работами В.Э.Карпова и Д.А.Добрынина (проект «Адаптант» [7]), в которых обучение осуществлялось методами:

- Обучение вероятностного автомата
- Нейронные сети
- Эволюционное моделирование

- ДСМ-метод

В отличие от других методов, ДСМ обеспечивает быстрое обучение – нужно малое количество обучающих примеров. Так, в остальных методах для обучения какому-либо рефлексу роботу требуется порядка 10-12 тактов обучения (для сравнения, 2 такта требуется в ДСМ-методе). Кроме того, ДСМ выдает полученные правила в явном виде, что облегчает интерпретацию результатов обучения. Из недостатков ДСМ-метода отмечается неприменимость в условиях противоречий в обучающих примерах. Идеи о том, как данная проблема могла бы быть решена, содержатся и в настоящей работе.

Элементы ИИ сейчас широко востребованы как применительно к малым мобильным роботам (например, бытовые роботы-пылесосы), так и к большим роботам (в частности, адаптивными являются известные шагающие роботы Big Dog и LittleDog компании Boston Dynamics).

Подробный обзор современных исследований в области интеллектуальных роботов изложен в статье [6]

Определение интеллектуального робота

Существует несколько определений интеллектуального робота. Например, такое: интеллектуальный робот должен выполнять задачу, сформулированную в общем виде ([7]).

В.К.Финн определяет ([12]) интеллектуального робота через понятие интеллектуальной системы, вводя иерархию (Когнитивную технологическую магистраль):

1 уровень – интеллектуальная система, данные вводятся и интерпретируются экспертом

2 уровень - когнитивная система (которая обладает перцепцией, то есть может получать информацию из окружающей среды),

3 уровень - интеллектуальный робот (который обладает также эффекторами, то есть могут влиять на окружающую среду).

Системы 1-го уровня существуют, тогда как системы 2 и 3 уровня ставят множество вопросов. Система 3 уровня должна обладать механизмом целеполагания, она должна выводить следствия из имеющейся ситуации, в ней должно учитываться время.

Данная работа – скорее попытка придумать систему 2 уровня. В ней рассматриваются вопросы о том, что делать с постоянно накапливающейся информацией: как её обрабатывать и хранить, что делать с возникающими противоречиями.

Цели работы

Целью работы является разработка компонентов самообучающейся интеллектуальной системы для робота.

Робот должен уметь получать факты из окружающей среды, обобщать их, и определять, как поступать в заранее неизвестной ему ситуации. Предполагается экспериментальное опробование созданной интеллектуальной системы на мобильном роботе.

ДСМ-метод

В качестве метода обучения мы использовали ДСМ-метод автоматического порождения гипотез, который обобщает в гипотезах информацию, полученную из обучающей выборки, и затем применяет эти гипотезы для классификации неизвестных объектов, а также имеет критерий достаточности гипотез.

Новизна предложенного решения

Отличия от реализации Д.А. Добрынина [5] заключаются в том, что: в созданной системе существуют как положительные, так и отрицательные обучающие примеры; введена ДСМ-стратегия, названная «ДСМ с исключаемыми примерами»; учителем является человек, а не алгоритм; фазы обучения и работы повторяются циклически.

Результаты

1. Создан ДСМ-решатель, пригодный для решения как общих задач, так и для решения поставленной задачи.
2. На его основе построен программно-аппаратный комплекс, включающий в себя интеллектуальную систему и мобильного робота.
3. Разработана ДСМ-стратегия «с исключаемыми примерами», пригодная для интеллектуального робота.
4. Экспериментальное опробование созданной интеллектуальной системы на мобильном роботе

Характеристика работы

Работа состоит из 3 глав, введения и заключения. В I главе изложены теоретические принципы, положенные в основу работы. Во II главе изложено устройство аппаратного комплекса, состоящего из управляющего компьютера и мобильного робота . В III главе описана архитектура системы с программной точки зрения.

Глава I. Теоретические сведения

I.1. Задача обучения робота

Для обучения робота естественно использовать один из методов *машинного обучения*. Робот может сам получать факты из окружающей среды, поэтому обучение должно быть *индуктивным*.

Задачей будет являться построение *индуктивного классификатора*, по модели «стимул-реакция» (условные рефлексы).

В качестве *обучающей выборки* возьмем совокупность пар «обстановка - поведение».

Под *обстановкой* понимается состояние окружающей среды, зафиксированное рецепторами робота – освещенность, цвет поверхности, наличие препятствия, угол по отношению к некоторому ориентиру (маяку) и проч.

Под *поведением* будем понимать влияние эффекторов робота на окружающую среду – вращение колес, свет, звук. Возможны более сложные поведения, состоящие из простых: например, поведение «Атака» является комбинацией езды вперед и света фонариком, а «Защита» - комбинацией езды назад и звука.

Обучение будет осуществляться на *примерах* (положительных и отрицательных), которые выдаёт учитель (человек-оператор).

Формирование положительного примера назовём *поощрением*, отрицательного примера – *наказанием.*

Обобщение нескольких примеров (только положительных или только отрицательных) назовём *гипотезой*.

Фазы *обучения* (обобщения, получения гипотез) и *действия* (применения гипотез) повторяются циклически.

Кроме того, для робота должен существовать некоторый признак, по которому он понимает, что обучение достаточно и теперь можно перейти к фазе работы.

Обучение является *интерактивным*, то есть сигналы поощрения и наказания выдаются непосредственно в ходе работы системы.

I.2. ДСМ-метод – краткое изложение

ДСМ-метод был предложен В.К.Финном (ВИНИТИ РАН) в конце 70-х гг. XX века. Метод назван в честь английского мыслителя Джона Стюарта Милля, чьи методы частично формализованы в ДСМ-методе. ДСМ – это логико-комбинаторный метод машинного обучения.

ДСМ-метод оперирует сущностями трех сортов:

- *Объектами* предметной области
- *Свойствами* этих объектов
- Возможными *причинами* этих свойств

Например, в области химии объектами могут быть химические соединения, свойствами – биологическая активность (токсичность), а причинами – фрагменты химических соединений.

Предполагается, что объекты имеют структуру. Обычно они состоят из элементов (*признаков*) и включают части (*фрагменты*). Предполагается, что причинами свойств объектов являются фрагменты их структуры.

На фрагментах должна быть определена *операция сходства* — бинарная операция, которая идемпотентна, коммутативна и ассоциативна (нижняя полурешетка).

Чаще всего фрагмент и свойства кодируются битовыми строками, а операция сходства определяется как побитовая конъюнкция.

Можно сказать, что объект – это кортеж, позиции соответствуют признакам , а фрагмент – это кортеж, некоторые позиции которого могут быть не заполнены. J-я компонента кортежа соответствует значению J-го

признака у описываемого объекта. Если J-я компонента фрагмента не определена, то это означает, что значение J-го признака не важно. Такой фрагмент может возникнуть, если он, например, является сходством объектов, у которых значение J-го признака различны.
Так, для робота пересечением фрагментов «есть свет, слева маяк» и «нет света, слева маяк» будет фрагмент «слева маяк».

Вводятся *внутренние* истинностные значения:
«+» - объект О обладает свойством Р (фактическая истина)
«-» - объект О не обладает свойством Р (фактическая ложь)
«t» - неизвестно, обладает ли О свойством Р (неопределенность)
«0» - сведения противоречивы (есть как доводы за, так и довод против)

Вводится понятие *примера*:

Объект о называется *положительным примером* (‘+’-примером) для свойства p, если мы знаем, что о обладает свойством p.

Объект о называется *отрицательным примером* (‘–’-примером) для свойства p, если мы знаем, что о не обладает свойством p.
Аналогично определяются ‘t’-примеры и ‘0’-примеры. Один тот же объект может выступать в роли положительного и отрицательного примера одновременно – по разным свойствам.

Задача ДСМ-рассуждения – доопределить неизвестные примеры, то есть вместо t поставить +1, -1 или 0.

ДСМ-рассуждение представляет собой *правдоподобный* вывод (из истинных посылок выводятся не обязательно истинные заключения). Оно является *синтезом познавательных процедур*: индуктивное обобщение (порождение гипотез), рассуждение по аналогии (предсказания), абдукция (принятие гипотез).

На шаге *индукции* происходит порождение гипотез о возможных причинах свойств.

Фрагмент с является возможной *причиной наличия свойства* p, если:

- с является общим фрагментом по крайней мере двух ‘+’-примеров для p;
- с не входит ни в один ‘–’-пример для p.

Фрагмент с является возможной *причиной отсутствия свойства* p (антипричиной), если:

- с является общим фрагментом по крайней мере двух ‘–’-примеров для p;
- с не входит ни в один ‘+’-пример для p.

На шаге *аналогии* происходит перенос свойств известных примеров, из которых порождена гипотеза, на неизвестный пример, включающий фрагмент этой гипотезы (доопределение неизвестных свойств на основе полученных гипотез).

Объект о *обладает свойством* p, если:

- содержит хотя бы одну возможную причину свойства p,
- не содержит ни одной возможной антипричины свойства p.

Абдукция по Ч.С.Пирсу:

D – множество фактов,

H – множество выдвинутых гипотез,

H объясняет D

Следовательно, гипотезы из H правдоподобны.

В ДСМ-методе гипотезы принимаются, если они объясняют исходные данные [11].

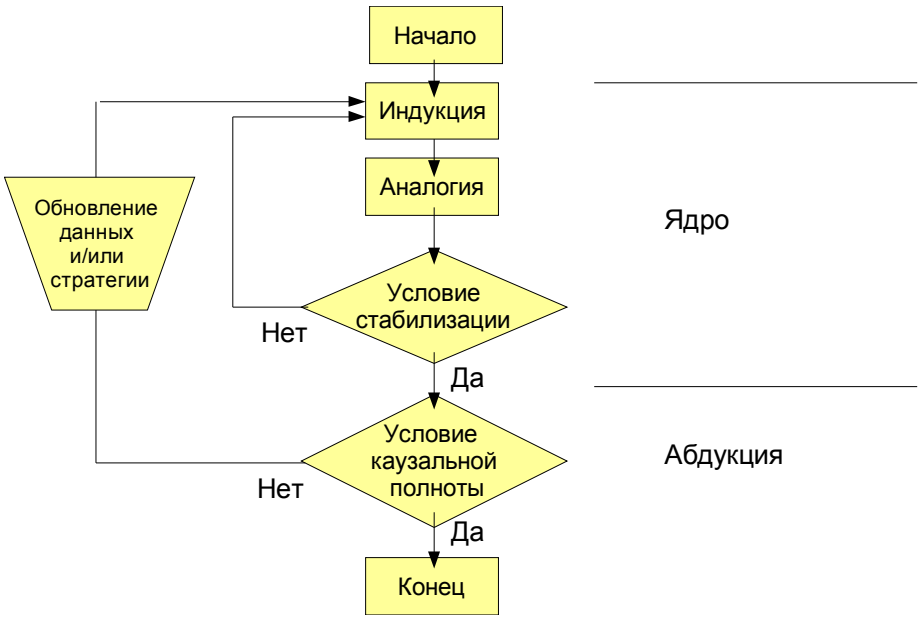

Рис.I.1. Блок-схема ДСМ-метода

Существует несколько *ДСМ-стратегий* (настроек):

- Простая без запрета на контрпример
- Простая с запретом на контрпример
- Обратный ДСМ
- Обобщенный ДСМ
- Несимметричный ДСМ

Можно вводить нижний *порог* числа контрпримеров; числа родителей у гипотезы.

Достоинства: логико-комбинаторный (не статистический) и хорошо работает на маленьких выборках

Недостатки: большая вычислительная сложность, необходимость задания операции сходства на объектах

Для описания и изучения ДСМ используется аппарат *многозначных логик*. С его помощью были строго доказаны некоторые утверждения о ДСМ-методе:

- Непротиворечивость
- Обратимость правил

- Единственность модели
- Выразимость в логике предикатов I порядка
- Дедуктивная имитация

На данный момент ДСМ-метод применяется в следующих областях: фармакология (выявление закономерностей вида “структура-активность”), медицина (диагностика заболеваний по симптомам), социология (выявление детерминант социального поведения), робототехника (обучение мобильного робота).

Данный краткий обзор был написан с использованием источников [1], [10]. Подробные сведения о ДСМ-методе, его теории и практическом применении изложены в сборниках статей [8] [3].

I.3. Метод Норриса

Для нахождения всех пересечений обычно используется *метод Норриса*. Он имеет линейную сложность от числа пересечений, и, в худшем случае, экспоненциальную от размера входа (2n). Достоинство метода Норриса в том, что он пошаговый, а не пакетный (при добавлении нового примера не нужно все пересчитывать заново)

На вход алгоритма подается множество уже порожденных пересечений, множество примеров, текущий пример.

Понятие каноничности пересечения нужно для того, чтобы избежать повторного вычисления пересечений, которые были вычислены ранее.

Пересечение является *каноническим*, если оно вычисляется путём последовательного пересечения примеров-родителей в возрастающем порядке.

Выделяют 2 вида каноничности – они соответствуют двум случаям и проверяются по-разному. Оба вида каноничности ведут к добавлению нового пересечения.

Пересекается ранее порожденное пересечение с текущим примером. *Относительная каноничность* – это отсутствие предыдущих примеров, фрагмент которых включает в себя пересечение фрагмента текущего пересечения с фрагментом текущего примера, при том, что этот предыдущий пример отсутствует в списке родителей текущего пересечения.

Проверяется итерацией по предыдущим примерам с фиксированным пересечением. В случае выполнения добавляется пересечение, фрагмент которого равен сходству фрагмента текущего примера и фрагмента текущего пересечения, а свойства – сходству их свойств.

Аналогично, для порождения одноэлементного пересечения вводится проверка абсолютной каноничности.

Абсолютная каноничность – если среди предыдущих примеров нет такого, в фрагмент которого вкладывается фрагмент текущего примера. Проверяется итерацией по предыдущим примерам для текущего примера. В случае выполнения добавляется пересечение, фрагмент и свойства которого равны фрагменту и свойствам текущего примера, а список родителей состоит из номера текущего примера.

Алгоритм:

Происходит итерация по уже порожденным пересечениям.

Для текущего пересечения проверяется, включается ли его фрагмент в фрагмент текущего примера.

Если да, то свойства пересечения заменяются на сходство его свойств со свойствами примера, а в список родителей добавляется номер примера.

Если нет, то происходит проверка на относительную каноничность, и, если она выполняется, добавляется новое пересечение.

Здесь заканчивается итерация по пересечениям.

Происходит проверка на абсолютную каноничность, и, если она выполняется, добавляется новое одноэлементное пересечение.

Данный алгоритм был изложен по источнику [2], который, в свою очередь, написан по материалам разработчиков ДСМ.

I.4. Решение задачи при помощи ДСМ-метода

Существующие ДСМ-системы работают с заранее заданным набором примеров. Примеры могут добавляться экспертом, если оказывается, что их недостаточно, но в этом случае происходит перезапуск системы. Для робота такой вариант неприемлем.

Д.А.Добрынин ввел понятие *динамического ДСМ-метода*, в котором примеры могут добавляться в процессе функционирования системы. [5]

Это накладывает требования на алгоритм поиска пересечений. Очевидно, что он должен быть *пошаговым*, а не *пакетным* - множество пересечений не должно полностью пересчитываться каждый раз, а должно лишь досчитываться при появлении новых примеров. Исходя из этого, был выбран алгоритм Норриса с некоторыми модификациями.

I.5. Постановка задачи «забывания»

Система управления робота не должна быть ресурсоемкой, так как бортовая ЭВМ компьютера имеет ограничения по размерам и потребляемой энергии. Робот – система реального времени. Следовательно, в памяти робота будут накапливаться знания, и будут там храниться, даже если устареют и потеряют актуальность (при смене окружающей среды). Кроме того, робот, как и человек, не может помнить всё – его память ограничена.
Неиспользуемые правила должны со временем «забываться» (их вес будет уменьшаться с каждым шагом). Это соответствует угасанию рефлекса (также предлагается в [7])

I.6. Веса

Для решения проблемы с ошибками в данных были попытки сделать следующий вариант:

Чтобы исключить случай, когда один контрпример уничтожает много раз подтвержденную гипотезу, гипотезам можно приписывать веса, которые берутся как число положительных примеров гипотезы. Тогда контрпример лишь делает гипотезу менее достоверной.

Когда роботу приходит новый пример, то, в случае наличия такого примера в базе фактов, вес гипотез, которые его объясняют, мог бы увеличиваться, а тех, которые ему противоречат – уменьшаться.

Однако, в ходе работы над текущим проектом не удалось придумать адекватную реализацию весов.

Глава II. Аппаратный комплекс

II.1. Технические характеристики робота

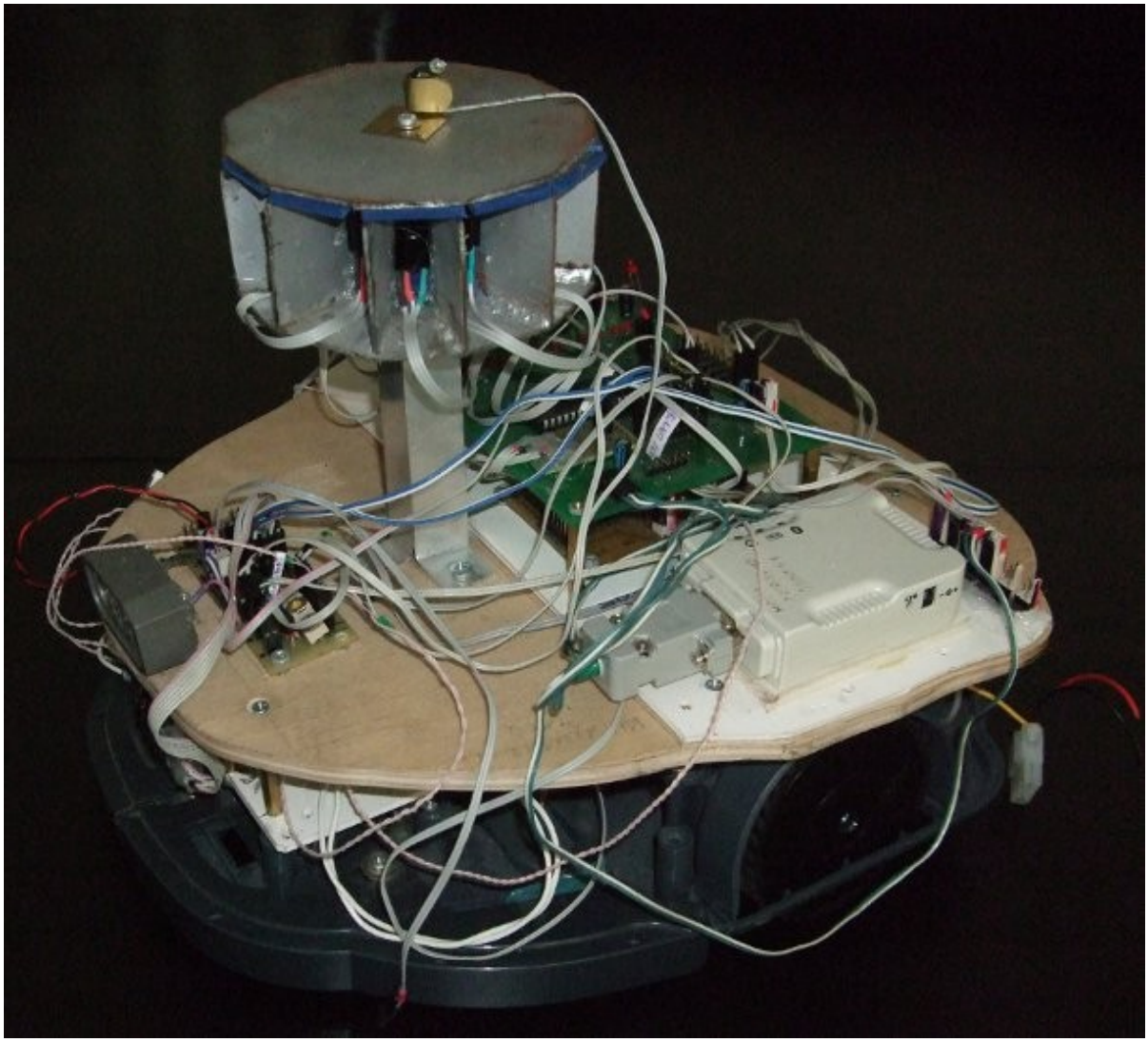

Рис.II.1. Внешний вид робота

Робот «Навигатор» представляет собой автономную тележку, имеющую на борту программируемый контроллер, набор датчиков (рецепторы), исполнительные механизмы (эффекторы), канал связи с управляющей ЭВМ. Ходовая часть взята от робота-пылесоса RV-2.

У робота имеется круговой приёмник ИК-сигналов, который определяет, под каким углом виден ИК-маяк. Программа фильтрации случайных отражений (например, от стен) заложена в бортовой микроконтроллер и описана в [9].

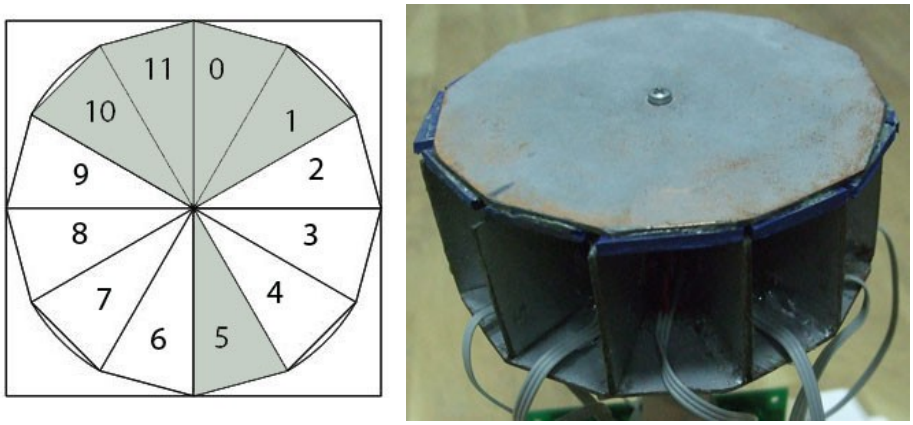

Рис.II.2. Круговая система приёмников

Бортовой микроконтроллер: ATMega 8 и ATMega 162 [13] [14], связанные по интерфейсу RS-485.

Рецепторы: 2 датчика линии, 2 ИК-датчика препятствий, 2 контактных датчика (бампера), 12 ИК-приемников, расположенных по кругу.

Эффекторы: Двигатели, фонарик, звук.

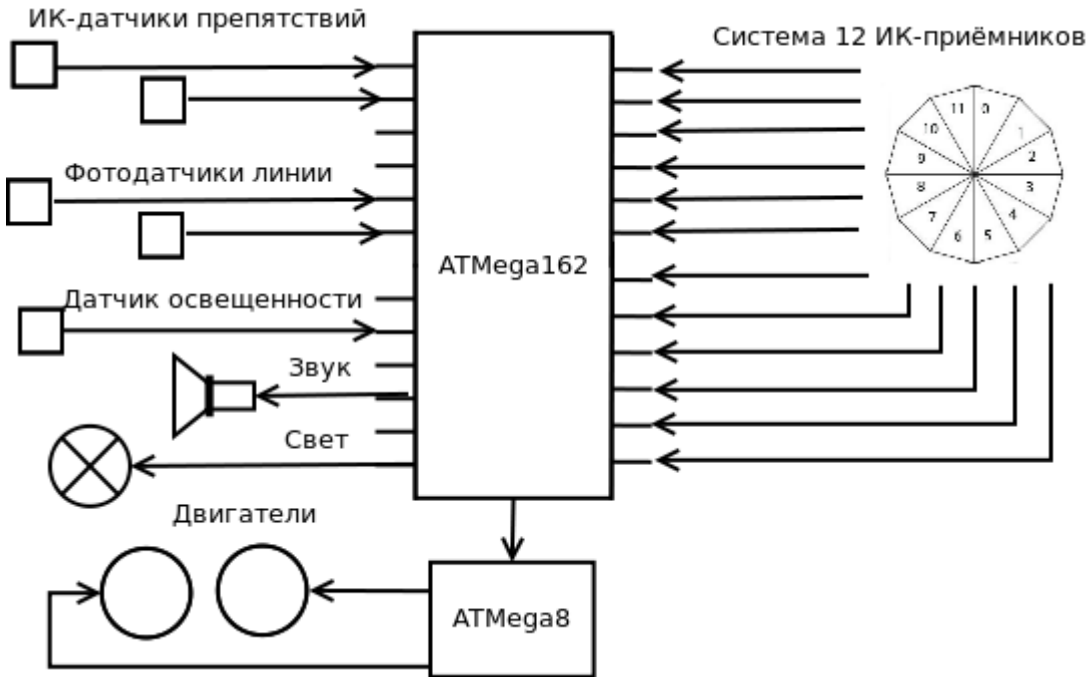

Рис.II.3. Бортовая система робота

Ввиду большого количества датчиков, был выбран микроконтроллер ATMega162. Он отличается тем, что имеет 30 линий ввода-вывода, из которых 22 были настроены как входы, а 8 как выходы. Недостаток в том, что входы только цифровые, т.е. микроконтроллер не имеет АЦП. Поэтому

для приёма аналоговых сигналов был выбран микроконтроллер ATMega8, который имеет 6 устройств АЦП в числе 13 линий ввода-вывода.

Было решено использовать ATMega8 в качестве подчинённого контроллера, обрабатывающего аналоговые сигналы. Практика показала, что вместо программной дискретизации сигналов проще использовать аппаратную (через эмиттерный повторитель), а получившийся цифровой сигнал заводить непосредственно на основной контроллер ATMega162.

Тем не менее, архитектура была оставлена прежней, и она допускает добавление новых датчиков. На данный момент подчинённый контроллер, следуя командам основного контроллера, управляет двигателями и выполняет ходовые функции, такие как следование по линии или управление при помощи джойстика.

II.2. Модельная задача

В мире робота существуют объекты, которые он учится классифицировать. Например, объектами могут быть совокупности параметров окружающей среды в некоторые моменты времени. У ситуации есть признаки – совокупность обстоятельств, и свойства – поведение робота в этой ситуации.

Для обучения робота была выбрана несложная модельная задача, которую можно было решить с имеющимся набором датчиков и исполнительных механизмов. Учитель обучает робота правилам вида «если - то» - например, «если светло и есть препятствие – входи в режим атаки», или «если темно и не видно маяка – входи в режим защиты».

Признаки:

- «Робот на черном/на белом»
- «Темно/светло»
- «Препятствие впереди/нет препятствия»
- «Видит маяк под углом a»

Свойства (сложные действия)

- Атака (Ехать вперед и светить фонариком)
- Защита (Ехать назад и пищать)
- Поиск (Повернуться влево на 90 градусов)

II.3. Протокол обмена данными

Был разработан асинхронный протокол обмена данными между роботом и компьютером. Робот постоянно отсылает управляющему компьютеру в закодированном виде сигналы на датчиках, а также исполняет команды, пришедшие с управляющего компьютера.

Связь осуществляется через COM-порт по интерфейсу RS-232, с Bluetooth на скорости 9600.

Структура посылки, состоящей из 4 байтов, изображена на рисунке:

Рис.II.4. Структура посылки, исходящей от робота

Так как датчиков, не относящихся к маяку, всего 7, то в 1-м байте посылки последний бит не имеет значения. Кроме того, во 2-м байте имеют значение только 5 битов: закодировано 24 угла, поэтому требуется максимум 5 битов (2^5=32). Однако, так как суммарный размер всё равно составляет более 1 байта, а пересылка осуществляется в байтах, было решено оставить схему заполнения такой (неоптимальной).

Скорость приёма ниже, чем скорость посылки, поэтому приходит сразу несколько посылок.

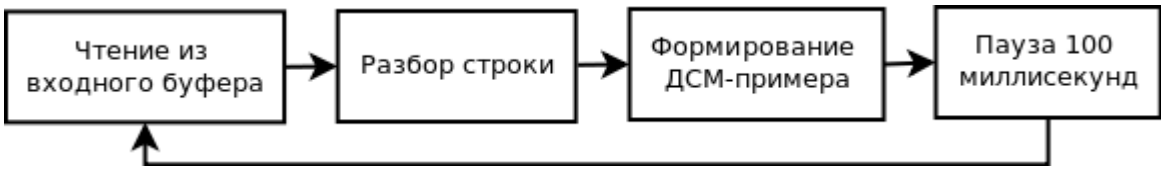

Рис.II.5. Приём и разбор посылки

Команды, посылаемые роботу, представляют собой 1 байт.

Глава III. Программный комплекс

III.1. Описание программной реализации

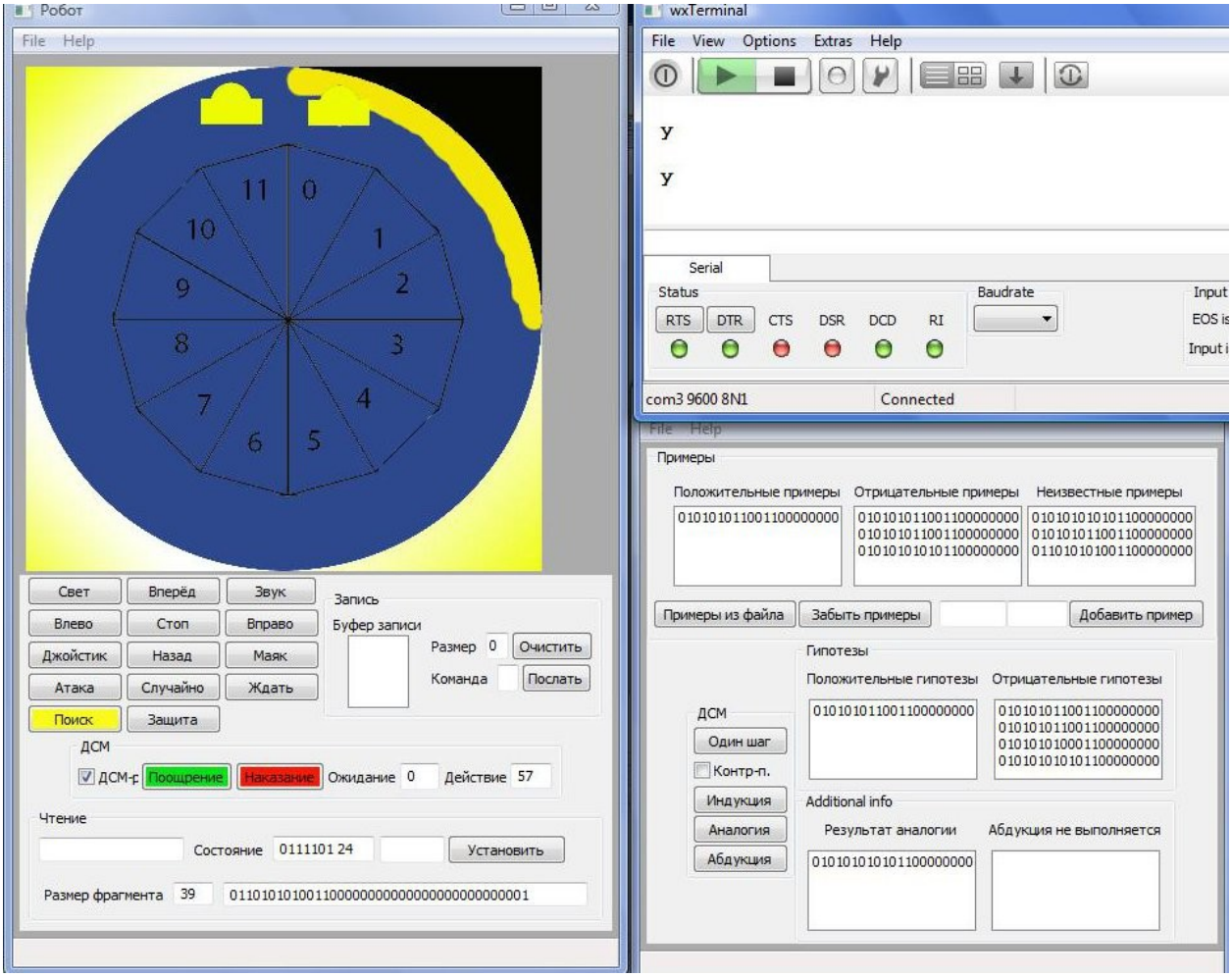

Рис.III.1. Графический интерфейс системы

В числе основных требований автором изначально были установлены:

1. Переносимость программы.
2. Лёгкость повторного использования кода.

Учитывая то, что иерархия классов ДСМ хорошо укладывается в парадигму объектно-ориентированного программирования, а также ввиду большого количества бибилиотек, был выбран язык C++. Были использованы только *свободные кроссплатформенные средства*.

Библиотеки:

- boost - класс битовых строк dynamic_bitset для представления данных в ДСМ-методе. Отличается от std::bitset тем, что размер определяется в ходе исполнения, а не компиляции. [15]
- ctb - использование и модификация терминальной программы wxterm [17] [18]
- wxWidgets - графический интерфейс [20]

Также из соображений общедоступности были выбраны среда разработки Code::Blocks и компилятор gcc. [16]

Для лёгкости повторного использования кода (в первую очередь – большой объём кода, относящегося к ДСМ-методу) было взято средство автоматического документирования: doxygen, которое порождает html-документацию из комментариев [19].

Таким образом, код ДСМ-метода тщательно документирован для облегчения повторного использования.

III.2. Программная реализация процесса обучения

В статье [7] вводится реализация схемы поощрения и наказания, при которой после окончания действия робот ждет оценки учителя некоторое время. В настоящей работе использован тот же принцип: есть *фазы* действия и ожидания.

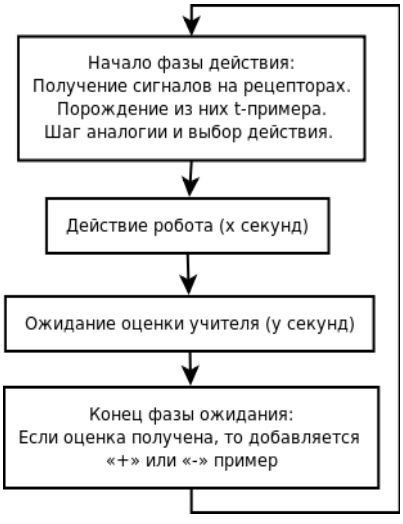

Рис.III.2. Схема обучения робота

Робот фиксирует обстановку и выбирает правило, которое подходит для данного случая; если такое правило не находится, то робот совершает случайное действие. Правильный поступок робота учитель поощряет. Если же робот ошибается, то оператор наказывает его (выдает отрицательный пример). Возможна реализация, при которой учитель предъявляет положительный пример (говорит, как нужно поступать в этой ситуации). Это нужно для того, чтобы робот с самого начала обучения мог набрать достаточное количество положительных примеров.

III.3. Формирование ДСМ-примеров

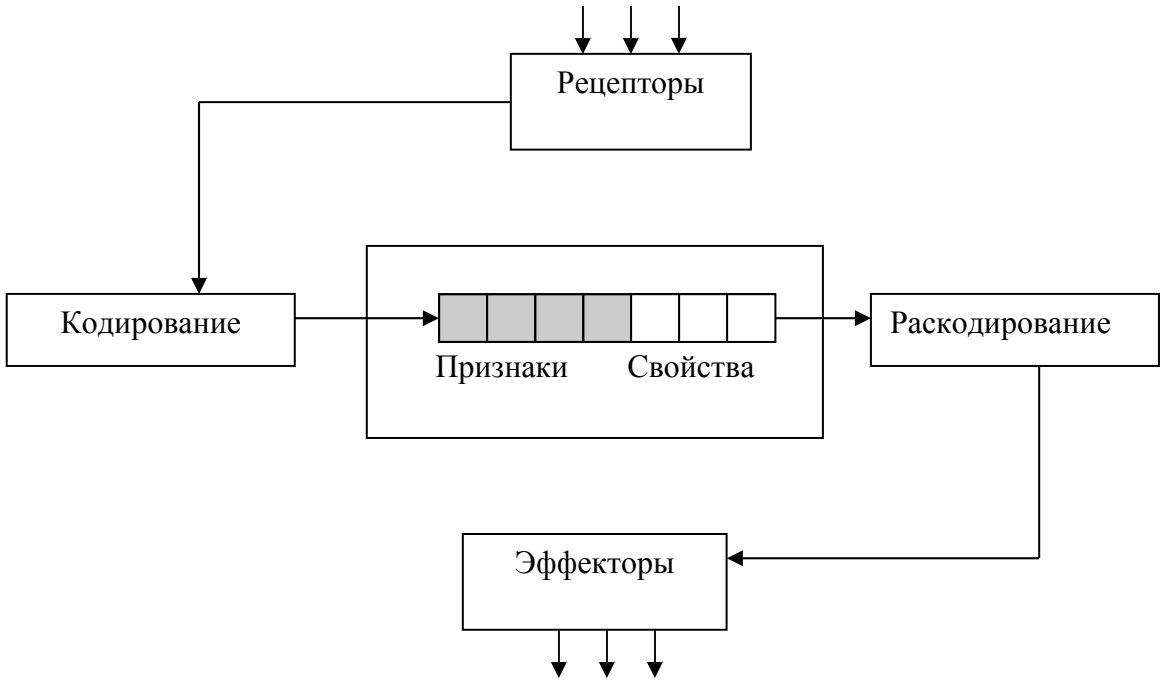

Рис.III.3. Связь робота с окружающей средой

Признаки являются закодированными сигналами на рецепторах робота.

В мире робота есть дискретные (наличие препятствия или наличие маяка) и непрерывные (освещённость окружающей среды) величины. Дискретизация происходит посредством сравнения с заранее заданными порогами. Например, величина «освещенность» преобразуется в признак «есть свет/нет света».

Важно, что при кодировании значение на датчике превращается в 2 ДСМ-признака: «есть сигнал» - «нет сигнала». Это нужно для того, чтобы обучаться ситуациям, когда отсутствие сигнала на датчике тоже является значимым.

Структура примера (признаки):

1. Левый бампер активен
2. Левый бампер неактивен
3. Правый бампер активен
4. Правый бампер неактивен
5. Левый ИК-датчик активен
6. Левый ИК-датчик неактивен
7. Правый ИК-датчик активен
8. Правый ИК-датчик неактивен
9. Датчик освещенности активен
10. Датчик освещенности неактивен
11. Левый датчик линии активен
12. Левый датчик линии неактивен
13. Правый датчик линии активен
14. Правый датчик линии неактивен
15. Угол на маяк 0 градусов
16. …
38. Угол на маяк 345 градусов
39. Маяк не виден

Структура примера (свойства):

1. Атака
2. Защита
3. Поиск

III.4. Архитектура системы

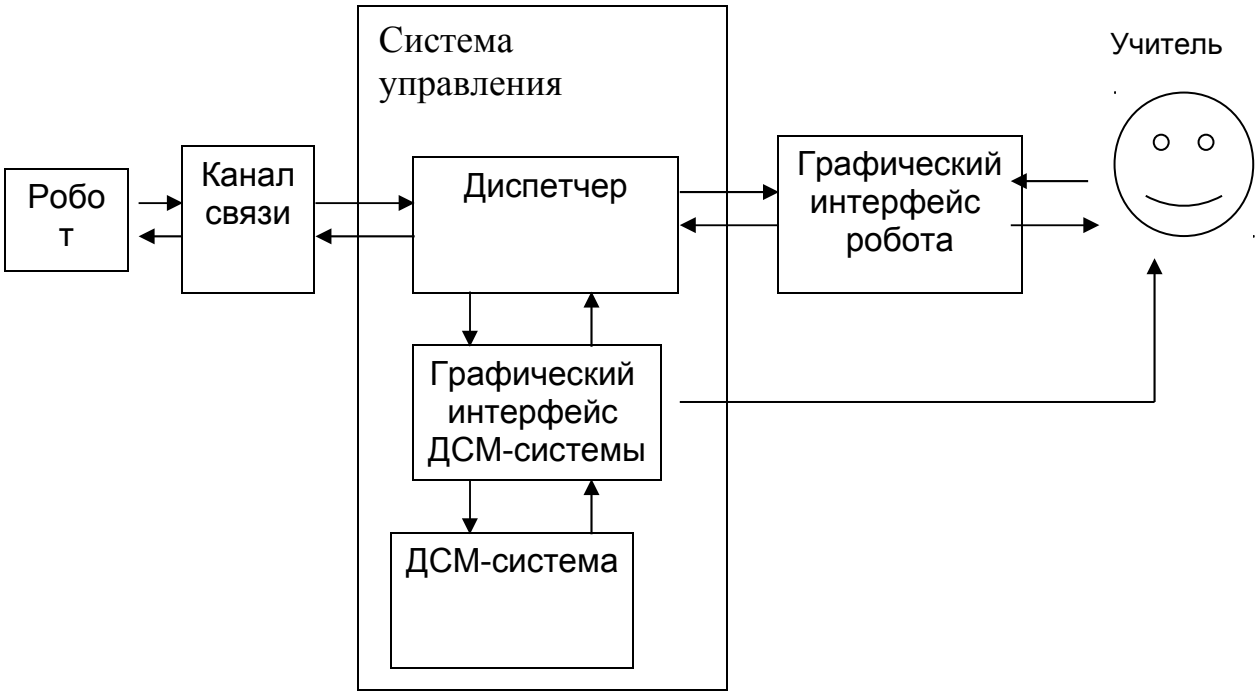

Рис.III.4. Архитектура системы

С программной точки зрения система состоит из 3 модулей, каждый из которых может быть запущен и отлажен отдельно:

1. Терминальная программа (для связи с роботом)
2. Графический интерфейс робота (для принятия сигнала от учителя)
3. ДСМ-система и ее графический интерфейс.

Особо хотелось бы также выделить программу-диспетчер, которая обеспечивает взаимодействие модулей.

Модули компилируются и отлаживаются по отдельности, затем в виде объектных файлов (.o) статически присоединяются к основной программе-диспетчеру.

III.5. Характеристика модулей и их взаимосвязь

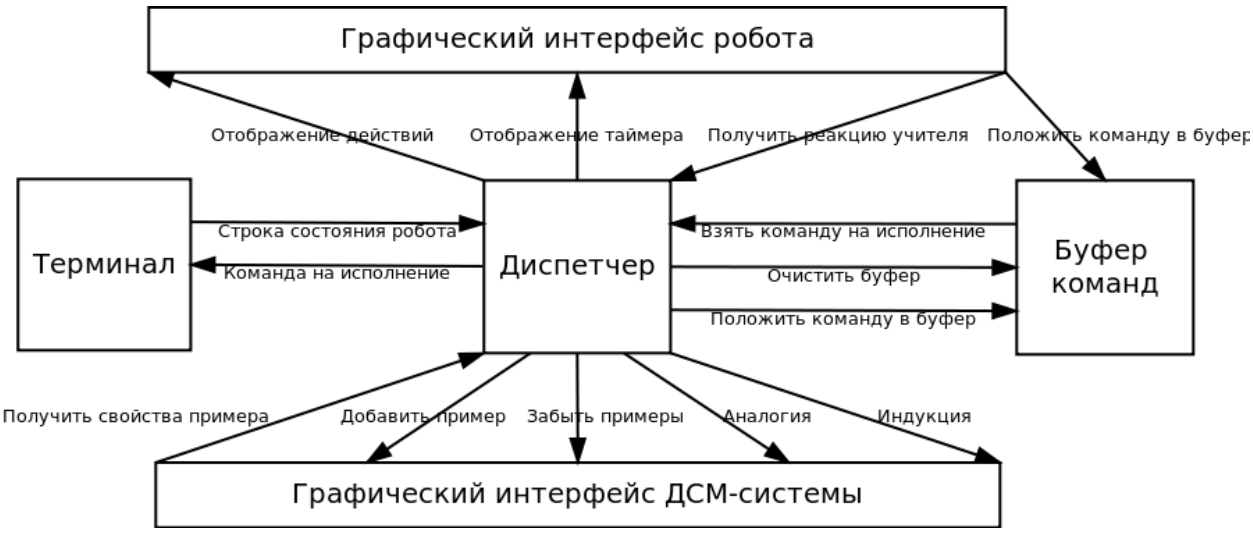

Рис.III.5. Взаимосвязь модулей

Терминальная программа (для связи с роботом)

- Осуществляет побайтовый приём информации и вывод на экран.
- Отсылает байты роботу

Графический интерфейс робота.

- Все команды, введённые пользователем, кладутся в буфер (это нужно для режима ручного управления роботом).
- Реакция учителя сохраняется в переменную
- Отображается текущее состояние робота
- Отображается (подсвечивается) текущее действие робота

Графический интерфейс ДСМ-системы. Выполняет двойную роль: служит как интерфейс ДСМ-системы, и является собственно графическим интерфейсом (обрабатывает ввод пользователя и отображает результаты)

- Отображает множества примеров, гипотез, результат аналогии и абдукции.
- Возвращает свойства примера с заданным фрагментом. Осуществляет поиск в контейнере «Результат аналогии»

- Выбор процедур ДСМ-метода (индукция, аналогия, абдукция, добавление примера, забывание примеров) и отображение их результатов.

Диспетчер

- Реализует алгоритм схемы обучения (см.III.2)
- Разбирает строку, приходящую от робота (вырезает последовательность между стоп-символами), и формирует строку состояния. Строка состояния отправляется в графический интерфейс робота для визуального отображения.
- Формирует строку ДСМ-фрагмента из строки состояния
- Формирует строку ДСМ-свойств из ответа учителя. Ответ учителя берется из графического интерфейса робота.
- Отправляет строку ДСМ-примера в графический интерфейс ДСМ-системы, где из нее уже формируется сам ДСМ-пример и добавляется в выборку.
- Раскодирует полученный из ДСМ-системы результат аналогии и кладет команды, соответствующие этому действию в буфер.
- Если на шаге аналогии не появилось «+» - свойств – выполняет случайное действие с учётом «-» свойств.
- Установка таймера
- Операции с буфером. Каждый раз команды берутся из буфера и отсылаются роботу через терминал. Код действия отправляется в графический интерфейс робота для визуального отображения. Буфер очищается.

III.8. Объектная модель ДСМ-метода

Модель устроена иерархически и соответствует парадигме объектно-ориентированного программирования. Примеры являются экземплярами

класса «объект», от этого же класса унаследован класс «пересечение».

Гипотезы представляются как указатели (точнее, итераторы) на пересечения.

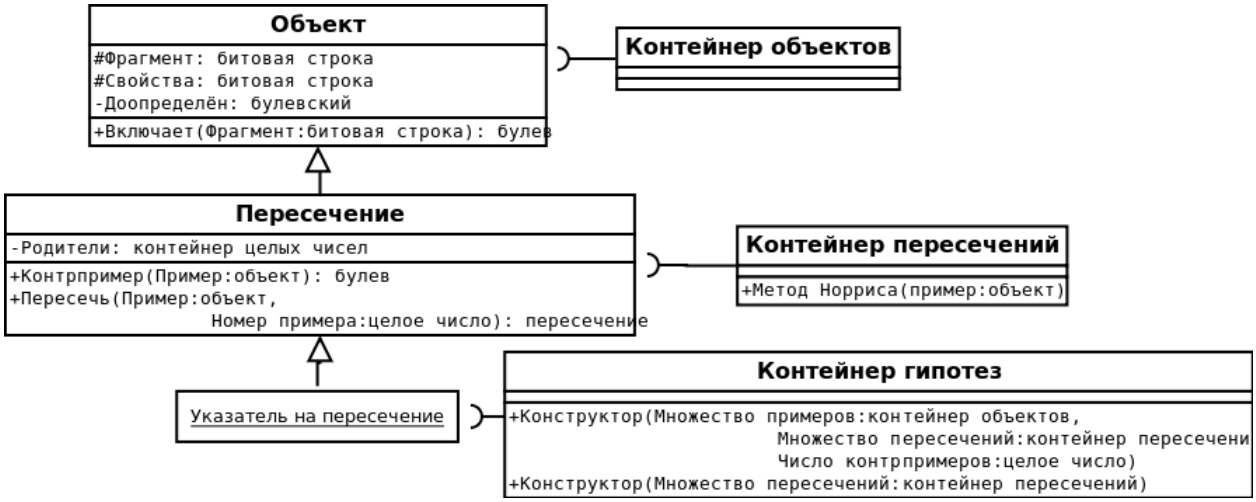

Рис.III.6. Иерархия классов

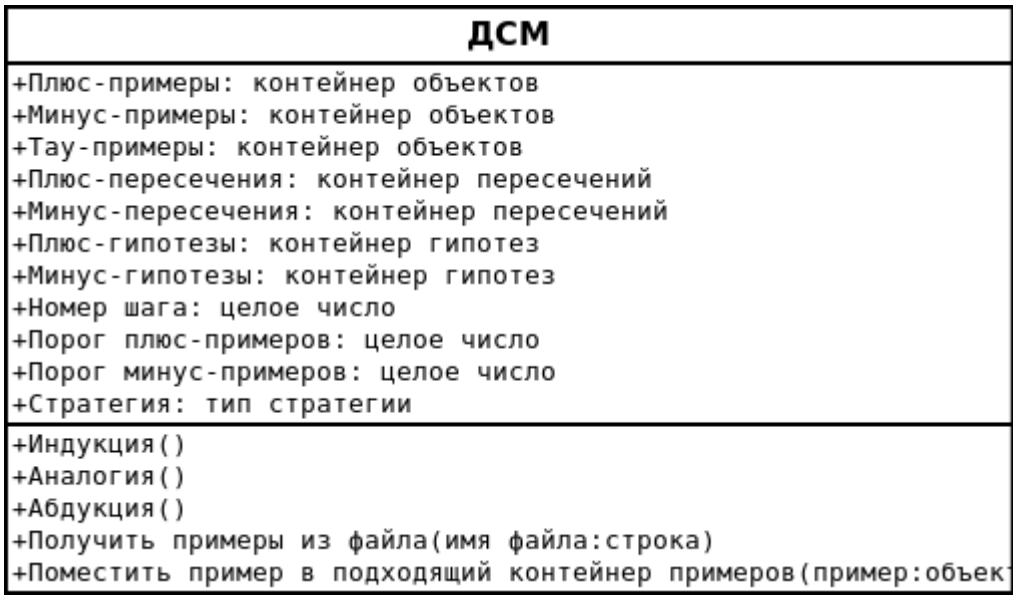

Рис.III.7. Класс «ДСМ»

Особенности:

- Гипотеза – это пересечение, которое удовлетворяет условиям: есть хотя бы одно «+» или «–» свойство, число родителей больше заданного порога (минимум - 2), число контрпримеров меньше заданного порога.
- Множество гипотез (в отличие от множества пересечений) каждый раз порождается заново, а не досчитывается, так как могут появиться новые контрпримеры.

- Классы-контейнеры унаследованы от стандартных контейнеров vector, и deque.

ДСМ-метод с исключаемыми примерами имеет некоторые отличия:

- Родители (примеры) из множества пересечений кладутся в контейнер объектов (используется полиморфизм), который затем используется как множество примеров на стадиях порождения гипотез (поиск контрпримеров) и абдукции. Родители – это пересечения, у которых один родитель, либо есть «скрытые родители».
- У класса «Контейнер объектов» появляется метод «Забыть n примеров», который удаляет n примеров с начала и увеличивает счетчик забытых примеров на n. Счетчик нужен, чтобы правильно выделять номера для новых примеров.

III.6. Постановка задачи «забывания»

Память робота ограничена, поэтому все примеры не могут храниться. Однако большинство их совпадает с пересечениями, у которых 1 родитель. Был разработан вариант ДСМ-метода (совместно с Д.В. Виноградовым), где примеры сохраняются как пересечения с одним родителем. После этого примеры могут быть забыты.

Для проверки каноничности итерация по примерам заменяется итерацией по пересечениям (недостаток - потенциально высокая вычислительная сложность). Однако, имеется выигрыш по объему занимаемой памяти: в классическом ДСМ в худшем случае мы имеем 2^n пересечений и 2^n примеров, тогда как в настоящей реализации – только 2^n пересечений, среди которых есть и примеры.

Таким образом, данный вариант может иметь смысл, когда признаков мало, а примеров много. У робота примеры непрерывно поступают, поэтому использование такой модификации оправданно.

Некоторые пересечения «скрывают» примеры: их фрагмент совпадает с фрагментом примера. Такие пересечения должны рассматриваться наравне с пересечениями с одним родителем. Практика показала, что число пересечений со «скрытыми родителями» не превосходит трети от всех.

III.7. Модифицированный метод Норриса

В проверке на относительную и абсолютную каноничность итерация по предыдущим примерам заменяется итерацией по пересечениям со «скрытыми родителями».

Отличия от классического метода Норриса выделены курсивом:

Происходит итерация по уже порожденным пересечениям.

Для текущего пересечения проверяется, включается ли его фрагмент в фрагмент текущего примера.

Если да, то свойства пересечения заменяются сходством его свойств со свойствами примера, а в список родителей добавляется номер примера. *Если фрагмент пересечения совпадает с фрагментом текущего примера, то у пересечения ставится флаг «есть скрытые родители».*

Если нет, то происходит проверка на относительную каноничность, и, если она выполняется, добавляется новое пересечение. *Если фрагмент пересечения совпадает с фрагментом текущего примера, то у пересечения ставится флаг «есть скрытые родители».*

Здесь заканчивается итерация по пересечениям.

Происходит проверка на абсолютную каноничность, и, если она выполняется, добавляется новое одноэлементное пересечение. *У пересечения ставится флаг «есть скрытые родители».*

Заключение

В результате работы были достигнуты следующие цели:

1. Создан ДСМ-решатель, пригодный для решения как общих задач, так и для решения поставленной задачи.

2. На его основе построен программно-аппаратный комплекс, включающий в себя интеллектуальную систему и мобильного робота.
3. Разработана ДСМ-стратегия «с исключаемыми примерами», пригодная для интеллектуального робота.
4. Экспериментальное опробование созданной интеллектуальной системы на мобильном роботе.

В данной работе были рассмотрены вопросы, возникающие при использовании ДСМ-метода в сфере робототехники. Очевидно, что простой перенос существующих ДСМ-систем был бы неэффективен из-за особенностей робота как системы реального времени.

Результаты работы могут быть использованы для наглядной демонстрации ДСМ-обучения в экспозиции Политехнического музея, в Лаборатории робототехники и искусственного интеллекта которого автором был построен робот.

В перспективе хотелось бы разработать схему весов и оценивания для гипотез, а также реализовать специфическую для робота абдукцию, что бы сделало робота ещё более интеллектуальным.

Литература

1. Аншаков О.М. Об одной интерпретации ДСМ-метода автоматического порождения гипотез. Научно-техническая информация. Сер. 2. Информационные процессы и системы / Всероссийский институт научной и технической информации РАН, 1999, № 1-2, с. 45-53
2. Ефимова Е.А. Алгоритм Норриса нахождения всех пересечений заданного набора множеств.
3. Автоматическое порождение гипотез в интеллектуальных системах. Сост Е.С. Панкратова, В.К. Финн; Под общ. Ред. В.К.Финна. Предисл. Ю.М. Арского – М.: Книжный дом «Либроком», 2009.

4. Гаазе-Рапопорт М.Г., Поспелов Д.А. От амебы до робота: модели поведения, М.: Наука, 1987, -286с.
5. Добрынин Д.А. Динамический ДСМ-метод в задаче управления интеллектуальным роботом. Десятая национальная конференция по искусственному интеллекту с международным участием КИИ-2006 (25-28 сентября 2006 г., Обнинск): Труды конференции. В 3-т. Т.2. - М: Физматлит, 2006. - 310 с.
6. Добрынин Д.А. Интеллектуальные роботы вчера, сегодня, завтра //X национальная конференция по искусственному интеллекту с международным участием КИИ-2006 (25-28 сентября 2006 г., Обнинск): Труды конференции. В 3-т. Т.2. М:Физматлит, 2006
7. Добрынин Д.А., Карпов В.Э. Моделирование некоторых форм адаптивного поведения интеллектуальных роботов. Информационные технологии и вычислительные системы №2, 2006 с.45-56 (http://raai.org/about/persons/karpov/pages/irobot/irobot.html)
8. ДСМ-метод автоматического порождения гипотез: Логические и эпистемологические основания. Сост. О.М. Аншаков, Е.Ф. Фабрикантова; Под общ. Ред. О.М. Аншакова. – М.: Книжный дом «Либроком», 2009.
9. Карпов В.Э., Волкова Т.А.. Управление роботом при движении по вектору (http://railab.ru/begin/labrab/mAzimuth.doc)
10. Подмаркова Н.О. Дидактическое FLASH-приложение по ДСМ-методу. //Десятая национальная конференция по искусственному интеллекту с международным участием КИИ-2006 (25-28 сентября 2006 г., Обнинск): Труды конференции. В 3-т., М: Физматлит, 2006
11. Финн В.К. Статья «Абдукция». Новая философская энциклопедия в 4-х томах. М.: Мысль. 2000-2001. (http://iph.ras.ru/page54852159.htm)
12. Финн В.К. Своевременные замечания о ДСМ-методе автоматического порождения гипотез //НТИ, сер.2 №8, 2009 (http://raai.org/about/persons/finn/pages/finn2009_JSM.doc)

13. http://www.atmel.com/dyn/resources/prod_documents/doc2513.pdf
Документация на микроконтроллер ATMega162.
14. http://www.atmel.com/dyn/resources/prod_documents/doc2486.pdf
Документация на микроконтроллер ATMega8.
15. http://www.boost.org/doc/ Библиотека boost.
16. http://www.codeblocks.org/user-manual Среда разработки Code::Blocks.
17. https://iftools.com/download/ctb/0.15/refman.pdf Библиотека ctb.
18. https://iftools.com/opensource/download.en.php Программа wxTerminal
19. http://www.stack.nl/~dimitri/doxygen/manual.html Средство автоматического порождения документации doxygen
20. http://www.wxwidgets.org/docs/ Библиотека wxWidgets