Робот для траектории на основе LEGO EV3

Эта задача является классической, идейно простая, она может решаться много раз, и каждый раз вы будете открывать для себя что-то новое.

Существует множество подходов для решения задачи следования по линии. Выбор одного из них зависит от конкретной конструкции робота, от количества сенсоров, их расположения относительно колёс и друг друга.

В нашем примере будет разобрано три примера робота на основе основной учебной модели Robot Educator.

Для начала, собираем базовую модель учебного робота Robot Educator, для этого можно использовать инструкцию в программном обеспечении MINDSTORMS EV3.

Так же, для примеров нам понадобятся, датчики света-цвета EV3. Эти датчики света, как никакие другие, наилучшим образом подходят для нашей задачи, при работе с ними, нам не придётся забоится о интенсивности окружающего света. Для этого датчика, в программах мы будем использовать режим отражённого света, при котором оценивается количество отражённого света красной подсветки датчика. Границы показаний датчика 0 - 100 единиц, для «отсутствия отражения» и «полного отражения» соответственно.

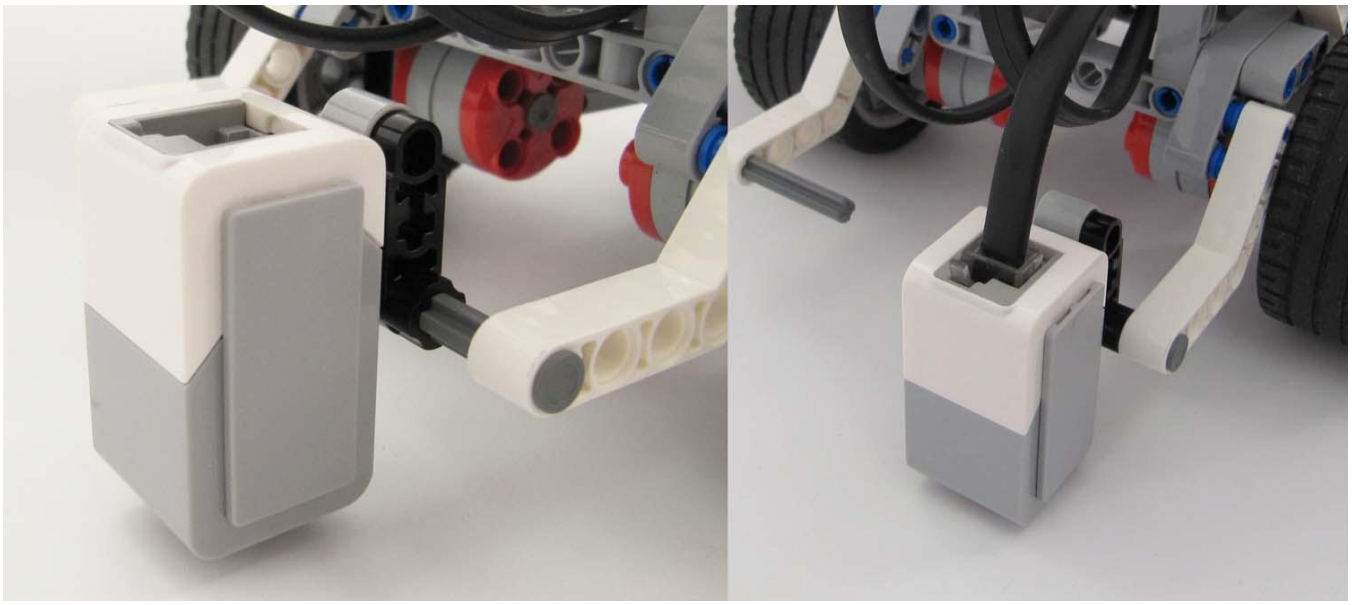
Для примера мы разберём 3 примера программ для движения по чёрной траектории изображённой на ровном, светлом фоне:

- Один датчик, с П регулятором.
- Один датчик, с ПК регулятором.
- Два датчика.

Пример 1. Один датчик, с П регулятором.

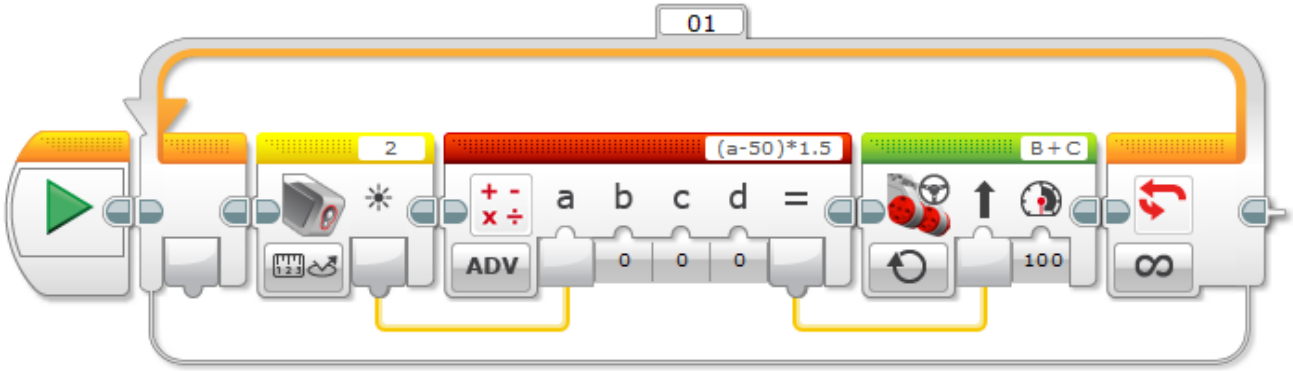
Конструкция

Датчик света устанавливается на балку, удобно расположенную на модели.



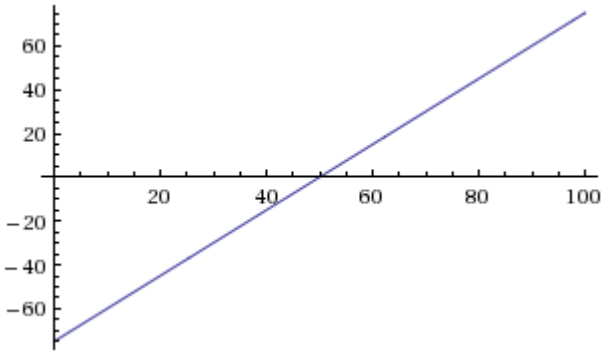
Алгоритм

Действие алгоритма основано на том, что в зависимости от степени перекрытия, пучка подсветки датчика чёрной линией, возвращаемые датчиком показания градиентно варьируются. Робот сохраняет положение датчика света на границе чёрной линии. Преобразовывая входные данные от датчика света, система управления формирует значение скорости поворота робота.



Так как на реальной траектории датчик формирует значения во всём своём рабочем диапазоне (0-100), то значением к которому стремиться робот, выбрано 50. В этом случае значения передаваемые функции поворота формируются в диапазоне -50 - 50, но этих значений недостаточно для крутого поворота траектории. По этому следует расширить диапазон в полтора раза до -75 - 75.

В итоге, в программе, функция калькулятора является простым пропорциональным регулятором. Функция которого (***(a-50)*1.5***) в рабочем диапазоне датчика света формирует значения поворота в соответствии с графиком:

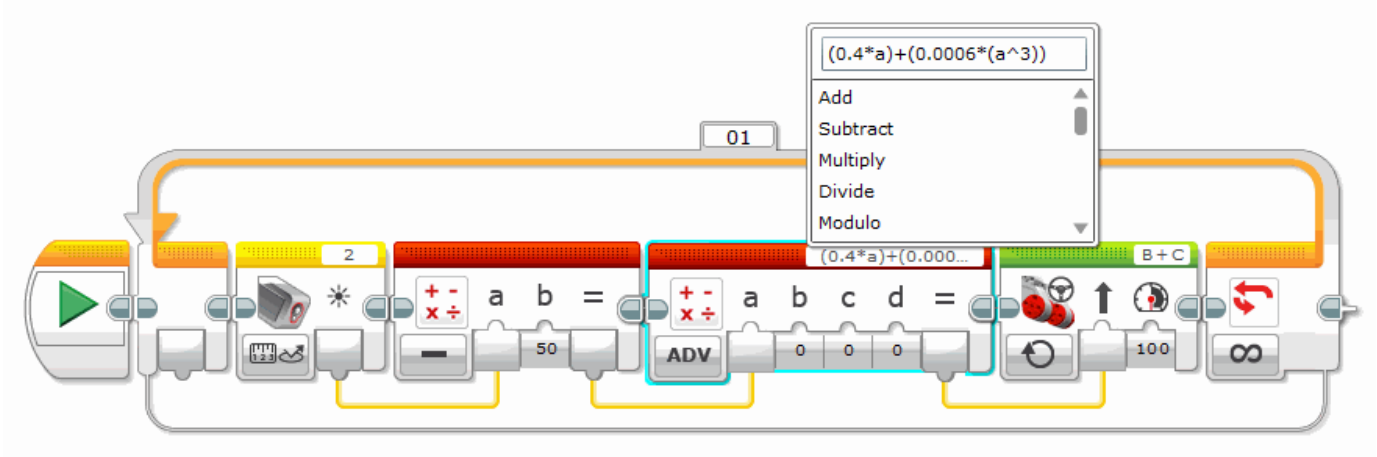


Пример работы алгоритма

Пример 2. Один датчик, с ПК регулятором.

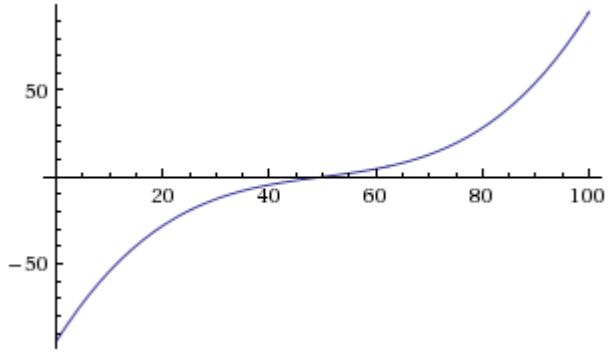
Этот пример составлен на той же конструкции.

Вы наверно заметили, что в прошлом примере робот излишне раскачивался, что не давало ему достаточно разогнаться. Сейчас мы постараемся немного улучшить эту ситуацию.



К нашему пропорциональному регулятору мы добавляем ещё и простой кубический регулятор, который добавит изгиб в функции регулятора. Это позволит уменьшить раскачивание робота рядом нужной границей траектории, а так же совершать более сильные рывки при сильном удалении от неё

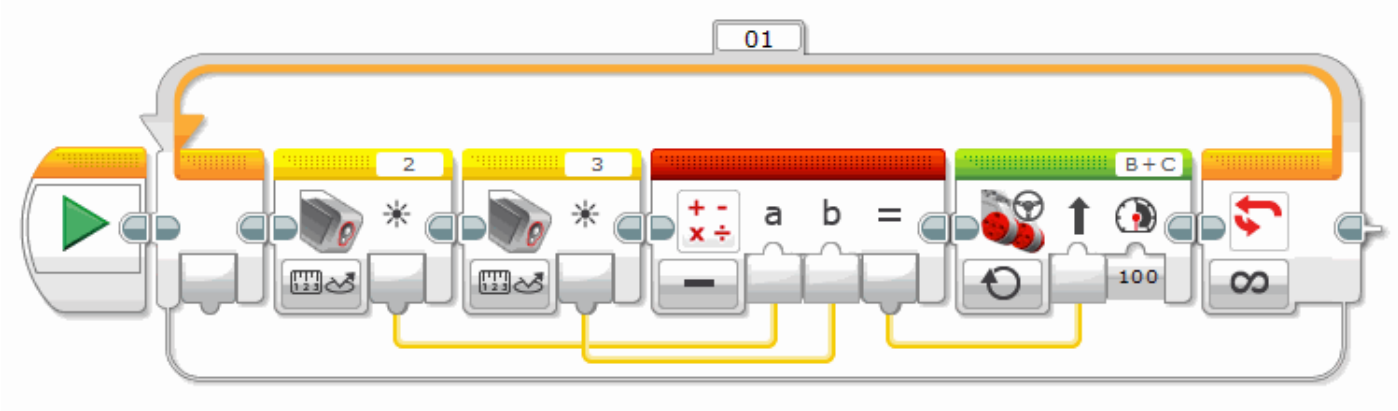
Если использовать настройку регулятора из примера, Пропорциональный коэффицнт 0.4 и Кубический коэффицент 0.0006 (***0.4*(a-50)+0.0006*((a-50)^3)***) то мы получим вот такую зависимость:



Пример работы алгоритма

Пример 3. Два датчика.

Использование двух датчикаов позволяет более чётко разграничить отклонение датчиков от линии и позволяет легко отфильтровывать/подсчитывать перекрёстки или сложные повороты на траектории.



Автор: Грудев Павел

http://wroboto.ru/rules/RobotExample/RobotExamples-component_54.html