Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Контрольная работа

УЧЕБНАЯ ЗАДАЧА МОДЕЛИРОВАНИЯ СИСТЕМЫ УПРАВЛЕНИЯ ТОРМОЖЕНИЕМ

МОБИЛЬНОЙ РОБОТИЗИРОВАННОЙ ПЛАТФОРМЫ

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | ст. Полховский А. Ф. |
|  |  | группа № 990541 |
|  |  |  |
| Проверил |  | проф. Татур М.М |

Минск 2023

**ВВЕДЕНИЕ**

При проектировании робототехнических комплексов типовой задачей является задача нелинейного управления исполнительными устройствами. Нелинейность может проявляться как следствие адаптивности принятия решений в тех случаях, когда система в реальном времени реагирует на свое состояние и/или на состояние внешней среды.

При изучении (или при проектировании) алгоритмов нелинейного управления возникает определенная проблема. Так, изучаемый алгоритм можно исследовать (или демонстрировать) только в динамике и только в условиях взаимодействия объекта управления с внешней средой. Но использовать реальный объект или реальную внешнюю среду, особенно на ранних стадиях проектирования, практически невозможно. Поэтому, применяют компьютерное моделирование с использованием современных универсальных систем, таких как MathLab, MathCAD и др., либо специально разработанных программ. При моделировании важно не только владеть аппаратом моделирования, но и корректно ставить модельные задачи. Именно этому посвящается настоящая работа.

1. **ПОСТАНОВКА ЗАДАЧИ**

Пусть платформа движется с начальной скоростью ***V0***, и на расстоянии ***S0*** от препятствия включается автомат торможения. Платформа должна плавно остановиться в непосредственной близости от препятствия. Алгоритм управления торможением ***А*** постоянно получает текущие значения скорости ***v*** и расстояния ***Sкон***, оставшегося до препятствия, и вычисляет усилие ***F***, прикладываемое к системе торможения (Рис.1).

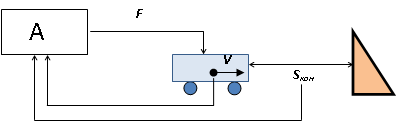


Рис. 1 – Мнемосхема учебной задачи на разработку алгоритма управления торможением

Необходимо:

1. С использованием средств моделирования (MatLab, Simulink или др.) разработать модель внешней среды» для решения учебной задачи торможения.
2. Разработать 3 альтернативных алгоритма торможения *а = f (v, Sкон).*
3. Выполнить моделирование разработанных алгоритмов, получить графики изменения скорости в исходном варианте постановки задачи.
4. Доработать модель «внешней среды», позволяющей:

* имитировать инерционность срабатывания исполнительных устройств;
* имитировать участок дороги с нулевым сцеплением;
* имитировать движение (на удаление) препятствия.

1. Выполнить моделирование разработанных алгоритмов в условиях п.4, получить графики изменения скорости в исходном варианте постановки задачи.
2. **МОДЕЛЬ ВНЕШНЕЙ СРЕДЫ**

Решение задачи следует начинать с построения модели внешней среды. На начальном этапе исследования, упростим модель и абстрагируемся от массы платформы, инерционных эффектов, инструментальных ошибок измерений параметров и др. факторов, объективно влияющих на процесс торможения. В идеальном случае будем полагать, что сила ***F*** и замедление, ***а*** – прямо пропорциональны, т.е. ***F∝а*** и мы, теоретически, можем «обеспечить» любое необходимое замедление.

При этом, обучаемым предлагается опираться на знания средней школы, т.е. оперировать категориями: скорость, расстояние, время, ускорение (замедление), а для расчета текущих значений ***v*** и ***Sкон*** в модели «внешней среды» использовать известные формулы замедленного движения. Тогда общую модель можно представить, как показано на рис. 2.

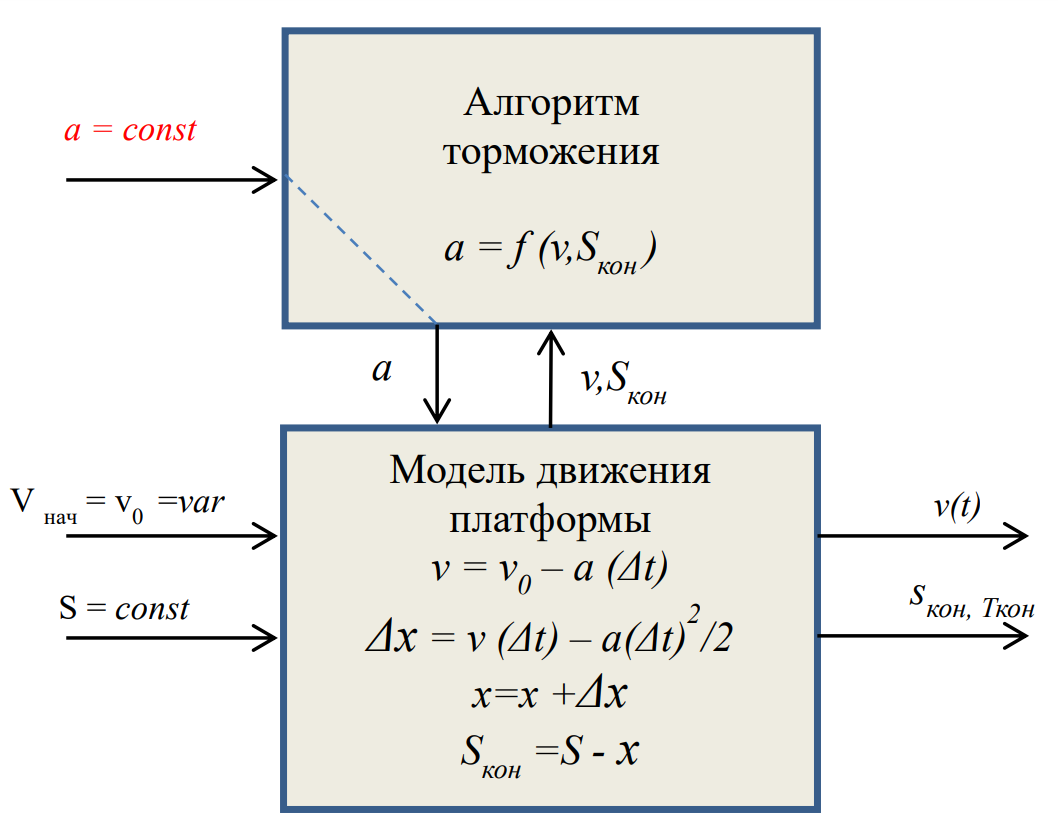


Рис. 2 – Схема упрощенной математической модели процесса торможения мобильной

На данном рисунке в явном виде модель разделена на две части. Первая – «Модель движения платформы», которая рассчитывает оставшееся расстояние до препятствия и текущую скорость, является моделью внешней среды. Вторая часть – представляет собой алгоритм управления торможением.

1. **АЛГОРИТМЫ УПРАВЛЕНИЯ ТОРМОЖЕНИЕМ**

Целью алгоритмов управления торможением является обеспечение стабильного и плавного замедления с минимальным расстоянием остановки.

В данном пункте будет рассмотрено несколько альтернативных алгоритмов управления торможением. Каждый из них имеет свои особенности и предназначен для определенных ситуаций. Разработка эффективных алгоритмов требует учета различных факторов, таких как скорость движения, расстояние до препятствия, состояние дорожного покрытия и другие внешние условия.

В рамках исследования будут рассмотрены три алгоритма управления торможением.

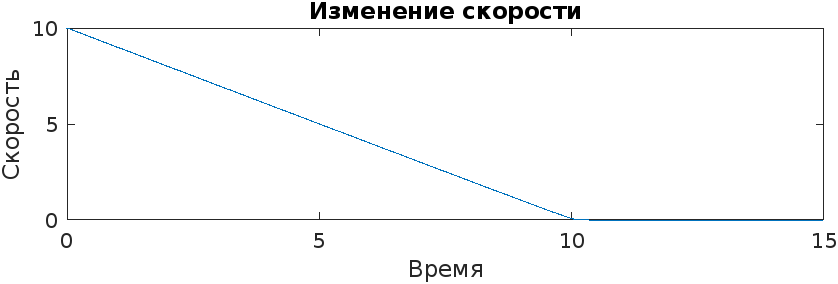
* 1. **Первый алгоритм. Линейное торможение с постоянным ускорением.**

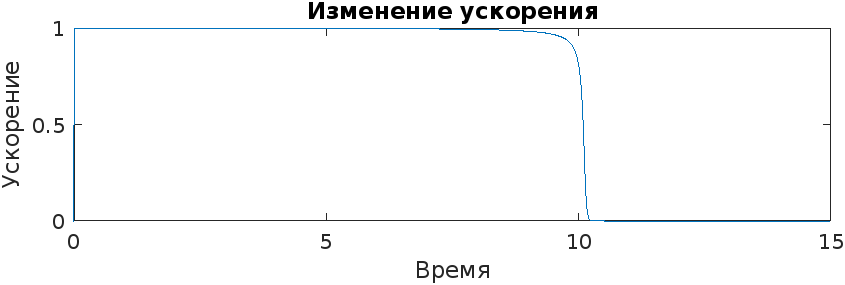
В этом алгоритме ускорение объекта пропорционально квадрату его текущей скорости и обратно пропорционально расстоянию до препятствия. Ускорение увеличивается по мере приближения объекта к препятствию, что приводит к усилению торможения.

Формула имеет следующий вид:

*a = v^2 / (2 \* Sкон);*

С использованием средств моделирования MatLab выполним моделирование работы алгоритма для получения графиков изменения скорости, расстояния, ускорения. Ниже полученные графики.





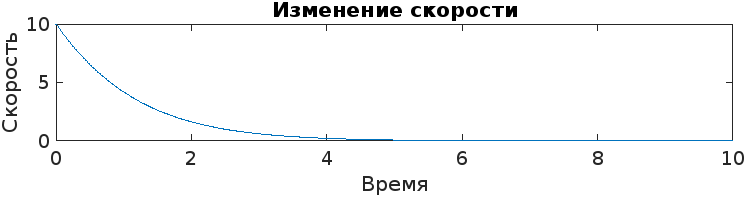
* 1. **Второй алгоритм. Экспоненциальное торможение с убывающим ускорением.**

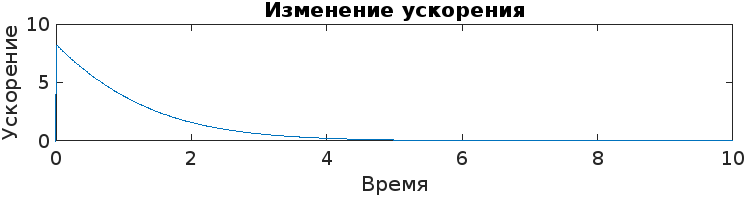
В этом алгоритме ускорение объекта экспоненциально убывает с увеличением скорости. Чем больше текущая скорость, тем меньше ускорение, что приводит к более плавному торможению по мере приближения к препятствию.

Формула имеет следующий вид:

*a = v \* exp(-v / Sкон);*

С использованием средств моделирования MatLab выполним моделирование работы алгоритма для получения графиков изменения скорости, расстояния, ускорения. Ниже полученные графики.





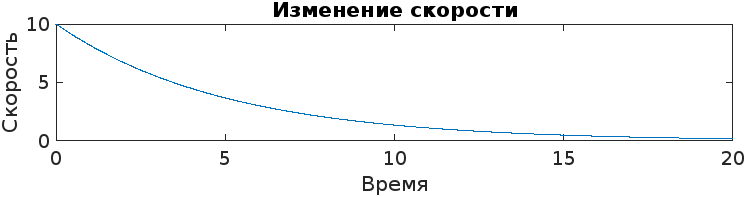
* 1. **Третий алгоритм. Квадратичное торможение.**

В этом алгоритме ускорение объекта пропорционально квадрату его текущей скорости и обратно пропорционально расстоянию до препятствия. Чем больше текущая скорость и чем меньше расстояние до препятствия, тем сильнее торможение. Этот алгоритм обеспечивает более интенсивное торможение при более высоких скоростях и близком расстоянии до препятствия.

Формула имеет следующий вид:

*a = v2 / Sкон;*

С использованием средств моделирования MatLab выполним моделирование работы алгоритма для получения графиков изменения скорости, расстояния, ускорения. Ниже полученные графики.



1. **УСЛОЖНЕНИЕ МОДЕЛИ ТОРМОЖЕНИЯ**

После того, как модель будет протестирована и отлажена, можно рассматривать более сложные варианты влияния внешней среды на торможение. Напомним, что в реальных условиях на замедление влияют такие факторы, как инерционность (масса) платформы, уклон, сцепление с поверхностью, время срабатывания исполнительных устройств системы торможения и др. Но в приведенной модели такие параметры как масса платформы, коэффициент сцепления с дорогой, процент уклона и т.п. не содержатся. Тем не менее, существует возможность косвенной имитации влияния названных факторов на замедление, например через введение времени запаздывания в алгоритм торможения и др. способы. В результате зависимость между прикладываемой силой ***F*** и замедлением ***a*** становится нелинейной.

Для таких, усложненных моделей внешней среды, алгоритмы управления должны проявлять адаптивность, т.е. «стремиться» остановить платформу не смотря на отрицательное влияние внешних факторов. И как раз в этих случаях выявляются сильные и слабые стороны разрабатываемых алгоритмов.

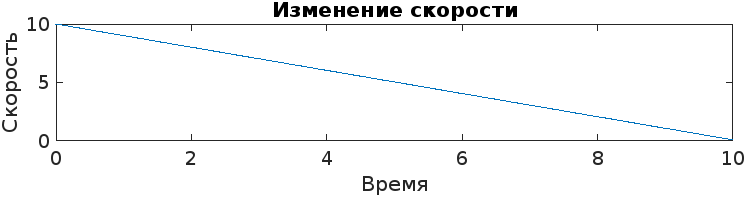
Доработаем модель внешней среды. Добавим возможности:

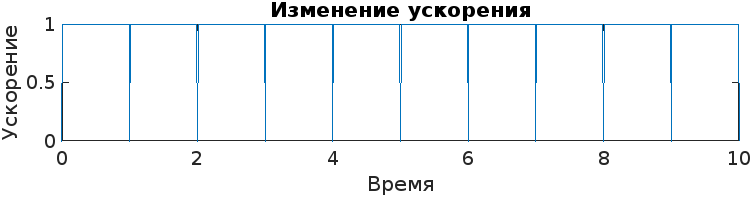
* имитировать инерционность срабатывания исполнительных устройств;
* имитировать участок дороги с нулевым сцеплением;
* имитировать движение (на удаление) препятствия.

Для доработки модели "внешней среды" с учетом инерционности срабатывания исполнительных устройств можно ввести задержку времени между моментом принятия решения об изменении ускорения и моментом его фактического применения. Это позволит имитировать задержку в работе исполнительных устройств и добавить инерционность в систему.

Добавим необходимые изменения в алгоритмы и выполним моделирование в среде MatLab для получения графиков.

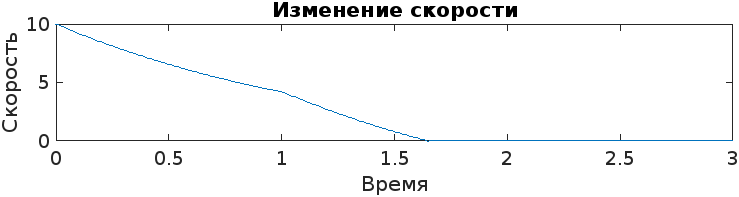
Полученные графики для алгоритма линейного торможения с постоянным ускорением:

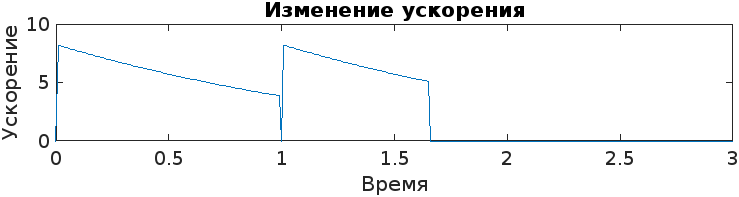




Исходя из результатов проведенного моделирования, мы можем сделать вывод, что несмотря на инерционность, данный алгоритм отлично справился с поставленной задачей. Такой вывод можно сделать после изучения графика скорости.

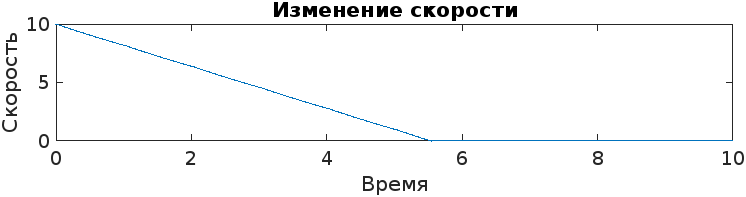
Полученные графики для алгоритма экспоненциального торможения с убывающим ускорением:

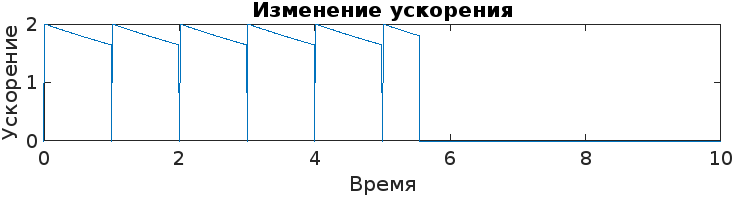




После проведения моделирования, мы можем изучить графики изменения скорости и ускорения. На графиках мы видим, что алгоритм хоть с запаздыванием, но отрабатывает общую динамику снижения скорости.

Полученные графики для алгоритма квадратичного торможения:



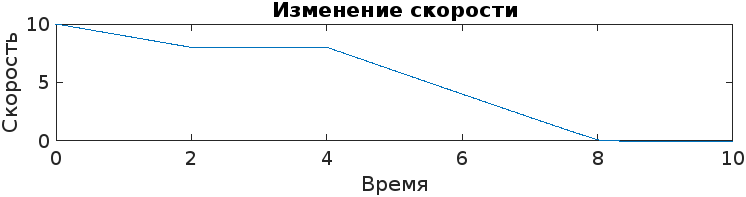


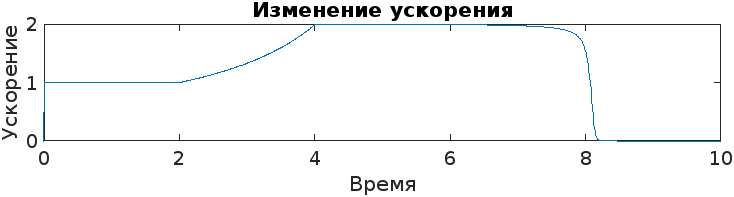
Изучая результаты проведенного моделирования, мы видим, что алгоритм смог снизить скорость, не взирая на инерционность срабатывания исполнительных устройств.

Для имитации участка дороги с нулевым сцеплением можно ввести переменную, которая будет учитывать влияние сцепления на скорость объекта. В случае нулевого сцепления, скорость будет оставаться на постоянном уровне, чтобы смоделировать ситуацию, когда объект теряет сцепление с дорогой.

Добавим необходимые изменения в алгоритмы и выполним моделирование в среде MatLab для получения графиков. Объект находится на участке с нулевым сцеплением с 2 до 4 секунды движения.

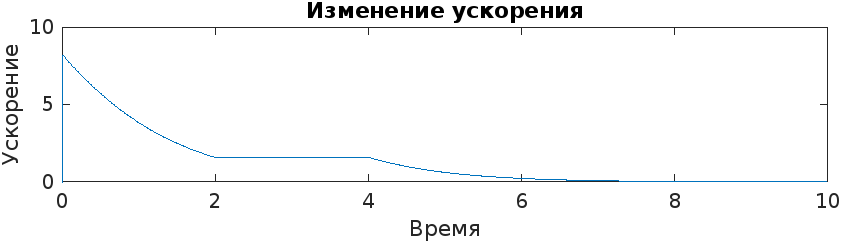
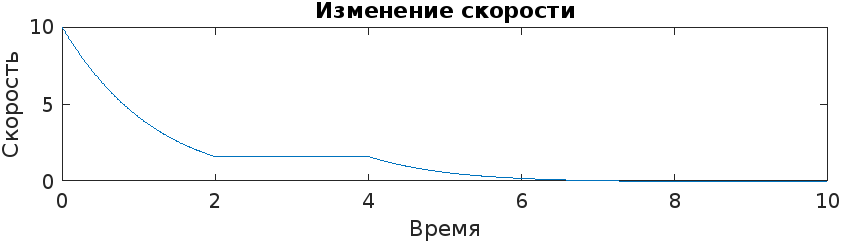
Полученные графики для алгоритма линейного торможения с постоянным ускорением:





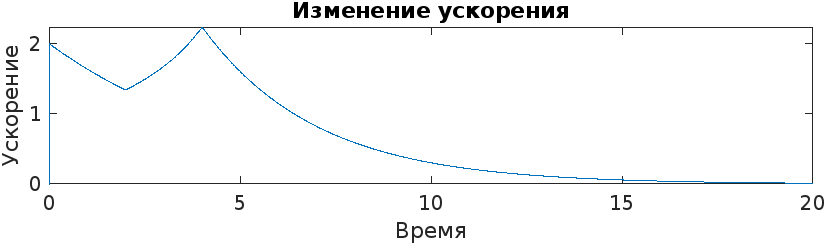
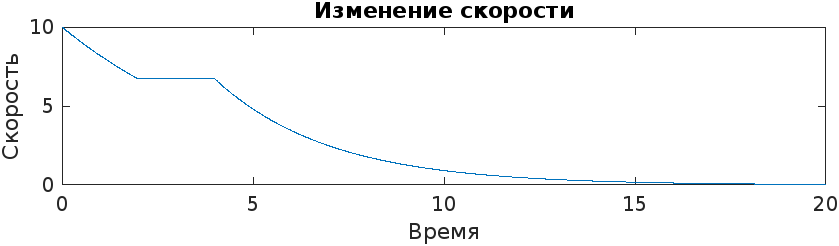
При изучении анализов проведенного моделирования, на графике скорости мы наблюдаем «стремление» алгоритма компенсировать отсутствие снижения скорости на скользком участке дороги.

Полученные графики для алгоритма экспоненциального торможения с убывающим ускорением:



Исходя из результатов проведенного моделирования, мы можем сделать вывод, что несмотря на нулевое сцепление, данный алгоритм не пытался компенсировать снижения скорости. Это увеличило время на торможения.

Полученные графики для алгоритма экспоненциального торможения с убывающим ускорением:

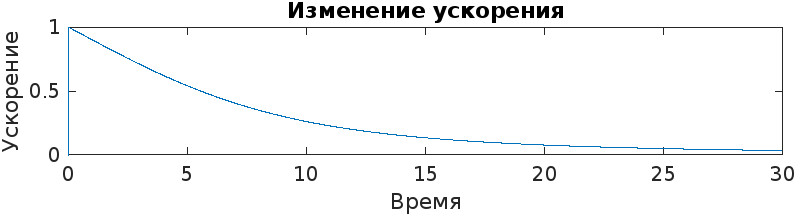
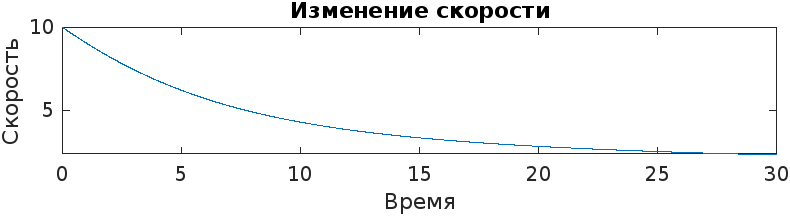


Изучая результаты проведенного моделирования, мы видим, что данный алгоритм, также как и алгоритм линейного торможения с постоянным ускорением, «старался» компенсировать отсутствие снижения скорости на скользком участке дороги.

Для имитации движения препятствия на удаление от платформы можно добавить изменение расстояния до препятствия с течением времени.

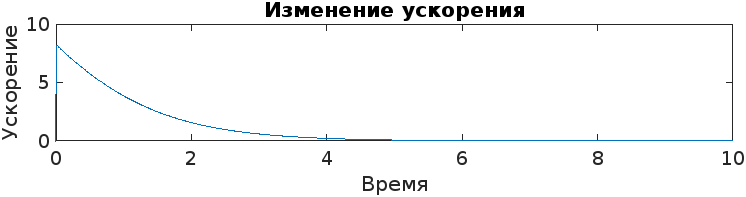
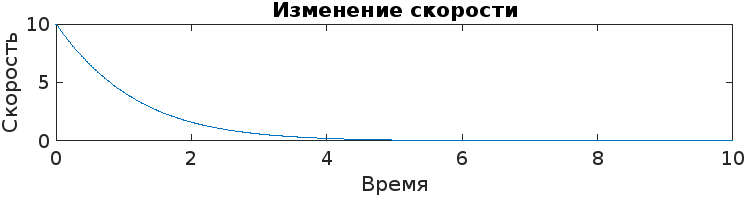
Добавим необходимые изменения в алгоритмы и выполним моделирование в среде MatLab для получения графиков.

Полученные графики для алгоритма линейного торможения с постоянным ускорением:



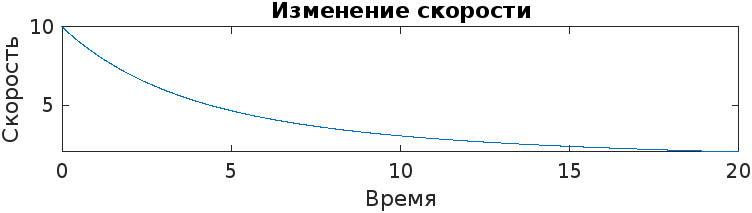
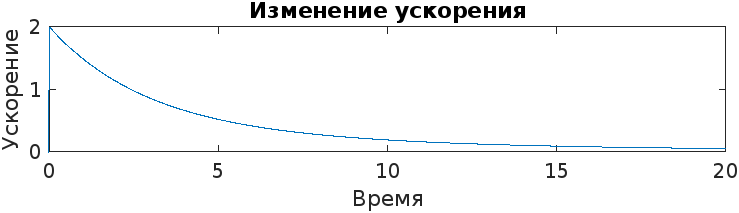
При изучении анализов проведенного моделирования, на графике скорости мы наблюдаем плавное торможение до препятствия. Из-за движения препятствия торможение заняло больше времени.

Полученные графики для алгоритма экспоненциального торможения с убывающим ускорением:



Исходя из результатов проведенного моделирования, мы можем сделать вывод, что данный алгоритм действовал схоже с алгоритмом линейного торможения. время на торможение увеличилось.

Полученные графики для алгоритма экспоненциального торможения с убывающим ускорением:



Изучая результаты проведенного моделирования, мы видим, что из-за движения препятствия на торможение потребовалось больше времени. В целом все три алгоритма на это усложнение модели отреагировали одинаково.

**ЗАКЛЮЧЕНИЕ**

При решении данной учебной задачи мы достигли главной цели: постигли методологию моделирования сложных систем, где есть объект моделирования и среда, в которой этот объект «живет» и проявляет свои свойства. Изучили среду моделирования MathLab.

Сделали вывод, что при моделировании важно корректно ставить задачу моделирования, чтобы ясно представлять какой результат моделирования ожидать. Любая задача моделирования, может быть, либо упрощена и выхолощена, либо усложнена до не реализуемости.

Поняли, что применение автоматизированных систем проектирования (моделирования) позволяет сократить трудоемкость решения поставленной задачи.

По аналогии с данной учебной задачей можно формулировать и решать другие задачи проектирования (моделирования).

**ПРИЛОЖЕНИЕ A**

**Листинг кода без усложнений**

% Начальные значения

V0 = 10; % начальная скорость

S\_kon = 50;

% Временные параметры моделирования

dt = 0.01; % шаг моделирования

t = 0:dt:20; % интервал времени

% Инициализация массивов для записи значений

v = zeros(size(t)); % скорость

S = zeros(size(t)); % расстояние

a = zeros(size(t)); % ускорение

% Задание начальных значений

v(1) = V0;

S(1) = S\_kon;

% Задание алгоритма торможения (выберите один из трех)

algorithm = 3;

for i = 2:length(t)

switch algorithm

case 1 %Линейное торможение с постоянным ускорением

% Расчет ускорения объекта

a(i) = v(i-1)^2 / (2 \* S\_kon);

case 2 %Экспоненциальное торможение с убывающим ускорением

a(i) = v(i-1) \* exp(-v(i-1) / S\_kon);

case 3 %Квадратичное торможение

a(i) = v(i-1)^2 / S\_kon;

end

% Расчет новой скорости объекта

v(i) = v(i-1) - a(i) \* dt;

% Расчет перемещения объекта

delta\_S = v(i) \* dt - a(i) \* dt^2 / 2;

S(i) = S(i-1) - delta\_S;

% Обновление расстояния до препятствия

S\_kon = S\_kon - delta\_S;

% Проверка условия остановки

if v(i) <= 0 && S\_kon >= 0

break;

end

end

% Визуализация результатов

figure;

subplot(3,1,1);

plot(t, v);

xlabel('Время');

ylabel('Скорость');

title('Изменение скорости');

subplot(3,1,2);

plot(t, S);

xlabel('Время');

ylabel('Расстояние');

title('Изменение расстояния');

subplot(3,1,3);

plot(t, a);

xlabel('Время');

ylabel('Ускорение');

title('Изменение ускорения');

**ПРИЛОЖЕНИЕ Б**

**Листинг кода с имитацией** **инерционности срабатывания исполнительных устройств**

% Начальные значения

V0 = 10; % начальная скорость

S\_kon = 50;

delay = 0.7; % задержка времени (инерция исполнительных устройств)

% Временные параметры моделирования

dt = 0.01; % шаг моделирования

t = 0:dt:20; % интервал времени

% Инициализация массивов для записи значений

v = zeros(size(t)); % скорость

S = zeros(size(t)); % расстояние

a = zeros(size(t)); % ускорение

% Задание начальных значений

v(1) = V0;

S(1) = S\_kon;

% Задание алгоритма торможения (выберите один из трех)

algorithm = 3;

for i = 2:length(t)

switch algorithm

case 1 %Линейное торможение с постоянным ускорением

% Расчет ускорения объекта

a(i) = v(i-1)^2 / (2 \* S\_kon);

case 2 %Экспоненциальное торможение с убывающим ускорением

a(i) = v(i-1) \* exp(-v(i-1) / S\_kon);

case 3 %Квадратичное торможение

a(i) = v(i-1)^2 / S\_kon;

end

% Задержка времени (инерция исполнительных устройств)

if i > delay/dt

a(i) = a(i - round(delay/dt));

end

% Расчет новой скорости объекта

v(i) = v(i-1) - a(i) \* dt;

% Расчет перемещения объекта

delta\_S = v(i) \* dt - a(i) \* dt^2 / 2;

S(i) = S(i-1) - delta\_S;

% Обновление расстояния до препятствия

S\_kon = S\_kon - delta\_S;

% Проверка условия остановки

if v(i) <= 0 && S\_kon >= 0

break;

end

end

% Визуализация результатов

figure;

subplot(3,1,1);

plot(t, v);

xlabel('Время');

ylabel('Скорость');

title('Изменение скорости');

subplot(3,1,2);

plot(t, S);

xlabel('Время');

ylabel('Расстояние');

title('Изменение расстояния');

subplot(3,1,3);

plot(t, a);

xlabel('Время');

ylabel('Ускорение');

title('Изменение ускорения');

**ПРИЛОЖЕНИЕ В**

**Листинг кода с имитацией участка дороги с нулевым сцеплением**

% Начальные значения

V0 = 10; % начальная скорость

S\_kon = 50;

% Временные параметры моделирования

dt = 0.01; % шаг моделирования

t = 0:dt:20; % интервал времени

% Инициализация массивов для записи значений

v = zeros(size(t)); % скорость

S = zeros(size(t)); % расстояние

a = zeros(size(t)); % ускорение

mu = zeros(size(t)); % коэффициент сцепления

% Задание начальных значений

v(1) = V0;

S(1) = S\_kon;

% Задание коэффициента сцепления

mu(t < 2) = 1; % участок дороги с нулевым сцеплением

mu(t >= 2) = 0; % остальная часть дороги

mu(t > 4) = 1;

% Задание алгоритма торможения (выберите один из трех)

algorithm = 3;

for i = 2:length(t)

switch algorithm

case 1 %Линейное торможение с постоянным ускорением

% Расчет ускорения объекта

a(i) = v(i-1)^2 / (2 \* S\_kon);

case 2 %Экспоненциальное торможение с убывающим ускорением

a(i) = v(i-1) \* exp(-v(i-1) / S\_kon);

case 3 %Квадратичное торможение

a(i) = v(i-1)^2 / S\_kon;

end

% Расчет новой скорости объекта

v(i) = v(i-1) - a(i) \* mu(i) \* dt; % Расчет новой скорости с учетом сцепления

% Расчет перемещения объекта

delta\_S = v(i) \* dt - a(i) \* dt^2 / 2;

S(i) = S(i-1) - delta\_S;

% Обновление расстояния до препятствия

S\_kon = S\_kon - delta\_S;

% Проверка условия остановки

if v(i) <= 0 && S\_kon >= 0

break;

end

end

% Визуализация результатов

figure;

subplot(3,1,1);

plot(t, v);

xlabel('Время');

ylabel('Скорость');

title('Изменение скорости');

subplot(3,1,2);

plot(t, S);

xlabel('Время');

ylabel('Расстояние');

title('Изменение расстояния');

subplot(3,1,3);

plot(t, a);

xlabel('Время');

ylabel('Ускорение');

title('Изменение ускорения');

**ПРИЛОЖЕНИЕ Г**

**Листинг кода с имитацией движения (на удаление) препятствия**

% Начальные значения

V0 = 20; % начальная скорость

S\_kon = 50;

obstacle\_v = 5;

% Временные параметры моделирования

dt = 0.01; % шаг моделирования

t = 0:dt:20; % интервал времени

% Инициализация массивов для записи значений

v = zeros(size(t)); % скорость

S = zeros(size(t)); % расстояние

a = zeros(size(t)); % ускорение

% Задание начальных значений

v(1) = V0;

S(1) = S\_kon;

% Задание алгоритма торможения (выберите один из трех)

algorithm = 2;

for i = 2:length(t)

switch algorithm

case 1 %Линейное торможение с постоянным ускорением

% Расчет ускорения объекта

a(i) = v(i-1)^2 / (2 \* S\_kon);

case 2 %Экспоненциальное торможение с убывающим ускорением

a(i) = v(i-1) \* exp(-v(i-1) / S\_kon);

case 3 %Квадратичное торможение

a(i) = v(i-1)^2 / S\_kon;

end

% Расчет новой скорости объекта

v(i) = v(i-1) - a(i) \* dt;

% Расчет перемещения объекта

delta\_S = v(i) \* dt - a(i) \* dt^2 / 2;

S(i) = S(i-1) - delta\_S + obstacle\_v\*dt;

% Обновление расстояния до препятствия

S\_kon = S\_kon - delta\_S + obstacle\_v\*dt;

% Проверка условия остановки

if v(i) <= 0 && S\_kon >= 0

break;

end

end

% Визуализация результатов

figure;

subplot(3,1,1);

plot(t, v);

xlabel('Время');

ylabel('Скорость');

title('Изменение скорости');

subplot(3,1,2);

plot(t, S);

xlabel('Время');

ylabel('Расстояние');

title('Изменение расстояния');

subplot(3,1,3);

plot(t, a);

xlabel('Время');

ylabel('Ускорение');

title('Изменение ускорения');