Міністерство освіти і науки України
Департамент освіти і науки Дніпропетровської облдержадміністрації
Дніпропетровське територіальне відділення МАН України

Відділення
технічних наук
Секція машинобудування та
робототехніки

ОБҐРУНТУВАННЯ ПАРАМЕТРІВ І РОЗРОБКА СИСТЕМИ КЕРУВАННЯ ГУСЕНИЧНИМ ТРАНСПОРТНИМ ЗАСОБОМ ІЗ НАВІСНИМ ПРИСТРОЄМ

Роботу виконав
Бараник Владислав Олександрович
учень 11 класу
середньої загальноосвітньої
школи № 11
м. Дніпропетровська

Наукові керівники:
Шибка Григорій Іванович, методист, керівник гуртка КПНЗ «МіськСЮТ» ДМР

Сірченко Артем Олександрович, науковий співробітник кафедри гірничих машин та інжинірингу Державного ВНЗ «НГУ»

Дніпропетровськ – 2015

ТЕЗИ

Тема наукової роботи «Обґрунтування параметрів і розробка системи керування гусеничним транспортним засобом із навісним пристроєм»; виконав Бараник Владислав Олександрович; Дніпропетровське відділення МАН України; гурток радіоелектроніки та мікропроцесорної техніки КПНЗ «Міськ СЮТ» ДМР; Державний вищий навчальний заклад «Національний гірничий університет»; учень 11 класу середньої загальноосвітньої школи № 11; м. Дніпропетровськ. Наукові керівники: Шибка Григорій Іванович, методист, керівник гуртка радіоелектроніки та мікропроцесорної техніки КПНЗ «Міськ СЮТ» ДМР; Сірченко Артем Олександрович, науковий співробітник Державного вищого навчального закладу «НГУ».

У роботі виконано **актуальне наукове завдання** обґрунтування параметрів і розробки системи керування гусеничним транспортним засобом із навісним пристроєм. **Мета роботи** – скласти рекомендації на проектування системи керування гусеничним транспортним засобом із навісним пристроєм. Для досягнення поставленої в роботі мети було виконано такі **завдання**: 1) аналіз стану питання і постановка завдань дослідження; 2) розробка тривимірної комп'ютерної моделі гусеничного транспортного засобу; 3) проведення обчислювального експерименту та аналіз отриманих результатів; 4) створення системи керування; 5) здійснення фізичного експерименту.

Було виявлено такі фактори: 1. При роботі гусеничного транспортного засобу за схемою «плавний поворот» величина кута його повороту лінійно залежить від маси навісного пристрою, причому ефективність процесу підвищується за умови відхилення навісного пристрою в бік центра кривизни траєкторії машини. 2. Коли гусеничний транспортний засіб працює за схемами «крутий поворот» і «поворот на місці», то оптимальне значення кута повороту досягається при розташуванні навісного пристрою на поздовжній осі машини. *Уперше було доведено*, що на траєкторію руху транспортного засобу з навісним пристроєм впливають такі чинники: схема бортового повороту, кут відхилення навісного пристрою від поздовжньої осі транспортного засобу, а також відносна маса навісного пристрою. Сформульовані на основі досліджень рекомендації було використано для створення системи керування гусеничним транспортним засобом із навісним пристроєм.

ЗМІСТ

ВСТУП

Актуальність теми. Машини з гусеничним рушієм досить різноманітні за конструкцією та призначенням. Це трактори, спеціальні тягачі, різні установки на гусеничному ходу, трубоукладачі на будівництві нафто- й газопроводів і т. д.

Відомо, що гусеничний рушій разом із системою керування ним – це один із найважливіших факторів впливу на тягові якості, тривалість експлуатації, економічність і надійність транспортних засобів.

У машинах з гусеничним рушієм найчастіше використовують навісне обладнання, причому властиве йому зміщення центру ваги ускладнює процес руху й маневрування транспортного засобу.

Технічна ідея роботи полягає у підвищенні ефективності керування машиною з гусеничним рушієм шляхом повороту навісного пристрою.

Очевидно, що, створюючи систему керування, необхідно враховувати механічні процеси, які виникають упродовж маневрування пристрою.

Отже, обґрунтування параметрів і розробка системи керування гусеничним транспортним засобом з навісним пристроєм на основі аналізу впливу його конструктивних параметрів на кут повороту є **актуальним науковим завданням**.

Тема дослідження має науковий і практичний характер, пов'язана з напрямом наукових розробок кафедри гірничих машин та інжинірингу Національного гірничого університету «Оптимізація параметрів механічного обладнання на основі сучасних методів комп'ютерного моделювання та обчислювального експерименту», а також з практичним напрямом створення моделей роботів, який реалізує «Астроклуб» – філія Дніпропетровської міської станції юних техніків.

Мета роботи – скласти рекомендації до проектування системи керування гусеничним транспортним засобом із навісним пристроєм.

Для досягнення поставленої в роботі мети належить виконати такі **завдання**:

1. Аналіз стану питання. Постановка завдань дослідження.
2. Створення комп'ютерної моделі гусеничного транспортного засобу.
3. Проведення обчислювального експерименту та аналіз отриманих результатів.

4. Створення системи керування.

5. Здійснення фізичного експерименту на реальній моделі.

Наукова ідея роботи полягає у застосуванні сучасних методів обчислювального та фізичного експерименту для встановлення залежностей між параметрами гусеничного транспортного засобу з навісним пристроєм.

Об'єкт дослідження – механічні процеси, які виникають у гусеничному транспортному засобі з навісним пристроєм при його повороті.

Предмет дослідження – залежності між параметрами гусеничного транспортного засобу з навісним пристроєм.

Методи дослідження – комп'ютерне моделювання поворотного руху машини засобами програмного забезпечення SOLIDWORKS, фізичне моделювання.

Робота має прикладний характер, у процесі її виконання було отримано перелічені нижче наукові результати.

Наукові положення

1. При роботі гусеничного транспортного засобу за схемою «плавний поворот» величина кута його повороту лінійно залежить від маси навісного пристрою, причому ефективність процесу підвищується за умови відхилення навісного пристрою в бік центра кривизни траєкторії машини.

2. Коли гусеничний транспортний засіб працює за схемами «крутий поворот» і «поворот на місці», то оптимальне значення кута повороту досягається при розташуванні навісного пристрою на поздовжній осі машини.

Наукова новизна: *уперше було доведено,* що на траєкторію руху транспортного засобу з навісним пристроєм впливають такі чинники: схема бортового повороту, кут відхилення навісного пристрою від поздовжньої осі транспортного засобу, а також відносна маса навісного пристрою.

Обґрунтованість і достовірність наукових положень висновків та рекомендацій забезпечено шляхом використання фундаментальних законів і методів механіки, ліцензійного програмного забезпечення SOLIDWORKS Education Edition (932B****3C7B****), розробки теоретичних моделей, адекватних експериментальним дослідженням (похибка результатів не перевищує 17 %).

Наукове значення роботи полягає в подальшому розвитку теорії гусеничного рушія і системи керування машиною з навісним пристроєм.

Практичну цінність роботи підтверджено рекомендацією автора стосовно розробки такої системи керування гусеничним транспортним засобом з навісним пристроєм, де було б враховано залежність між схемою бортового повороту і величиною кута його відхилення та масою навісного пристрою.

Реалізація результатів. Розроблені автором рекомендації було використано при створенні системи керування в діючій моделі гусеничного транспортного засобу з навісним пристроєм.

Апробація результатів. За результатами роботи було зроблено доповіді на XIII Всеукраїнській науково-технічній конференції «Потураївські читання», яка відбулася 19–20 січня 2015 р. в НГУ. Тези роботи буде опубліковано в матеріалах конференції.

Особистий внесок автора. Усі наукові ідеї, використані в роботі, наукові та практичні результати належать автору.

РОЗДІЛ 1
АНАЛІЗ СТАНУ ПИТАННЯ. ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

Серед машини з гусеничним рушієм – різноманітні за конструкцією та призначенням машини. Це, зокрема, трактори, спеціальні тягачі, різні установки на гусеничному ходу, трубоукладачі, задіяні в будівництві нафто- й газопроводів.

У літературних джерелах [1, 2, 3] розглянуто теоретичні засади роботи гусеничного рушія, зокрема питання статики, кінематики й динаміки взаємодії гусениць з напрямними й опорними котками та ведучим колесом, обґрунтовано стійкість обводу, процеси кочення опорного котка по рівній основі, втрати потужності в рушії та взаємодії його опорної гілки з ґрунтом (рис. 1.1).

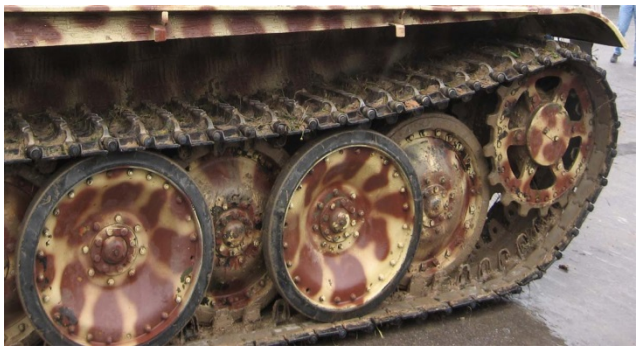

Рис.1.1. Загальний вигляд ділянки ходової частини танка

У згаданих роботах відзначено, що гусеничний рушій разом із системою керування ним – це один із найважливіших факторів впливу на тягові якості, тривалість експлуатації, економічність і надійність транспортних засобів.

Було встановлено можливість кількох схем бортового повороту машини, їх можна охарактеризувати таким чином [2]: плавний поворот, реалізований за допомогою відключення однієї з гусениць та її наступного короткочасного гальмування (рис. 1.2, *а*); крутий поворот, його досягають шляхом блокування однієї з гусениць (рис. 1.2, *б*); поворот на місці навколо центра мас (точки *О*) за допомогою перемикання приводу однієї з гусениць на задній хід (рис. 1.2, *в*). У схемах використано такі позначення: *V*л, *V*п – швидкість лівої і правої гусениць; *R* – радіус повороту; *B*т – база ТЗ.

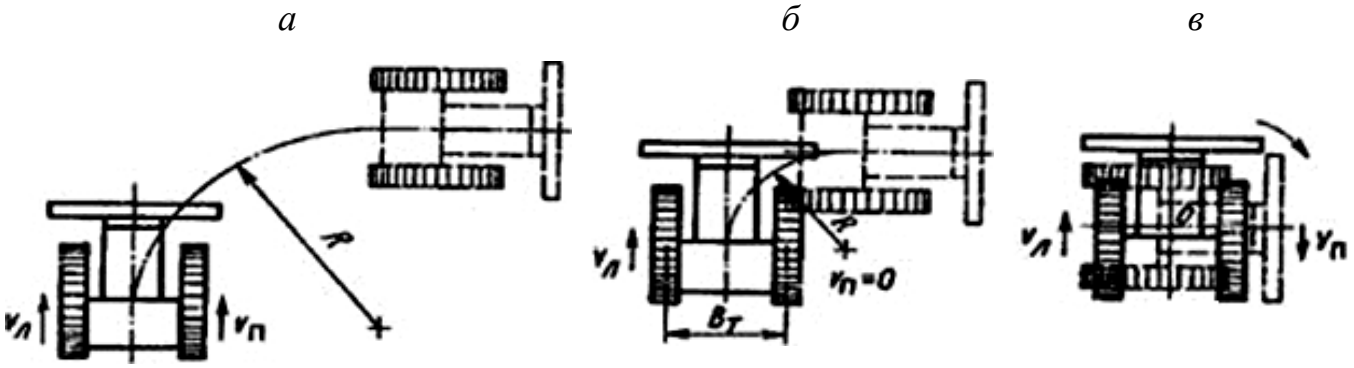

Рис.1.2. Схеми варіантів повороту транспортного засобу

Машини з гусеничним рушієм нерідко оснащують навісним обладнанням, причому під впливом властивого цим пристроям зміщення центру ваги ускладнюється процес руху та маневрування транспортного засобу.

Автори вищезазначених праць залишили поза увагою вплив на процеси повороту транспортного засобу (ТЗ) положення навісних пристроїв (НП).

Виконаємо аналіз перебігу механічних процесів у гусеничному рушії при повороті машини. Вважаємо, що приводи ведучих котків правої та лівої гусениць не залежать один від одного. Відомо, що поворот машини може відбуватися при різних швидкостях руху гусениць. Це супроводжується утворенням пари сил тертя, яка прагне повернути машину, і пари сил тертя, що перешкоджає повороту.

На рис. 1.3 зображено схему машини з гусеничним рушієм, обладнану навісним пристроєм. У схемі використано такі позначення: Gтз – маса ТЗ, Gнп – маса НП, Fзч, Fтер – сили тертя, що виникають при русі ТЗ в поздовжньому й поперечному напрямках, a, b – база ТЗ, α – кут відхилення НП.

Очевидно, що, розробляючи систему керування для машин з гусеничним рушієм, необхідно брати до уваги механічні процеси, що виникають при повороті навісного пристрою. У той же час процес повороту гусеничного рушія можна описати тільки у вигляді складних нелінійних моделей.

За таких умов у дослідженні механічного руху гусеничного рушія належить задіяти сучасні CAE-інструменти, які дозволяють віртуально відтворити механічний рух.

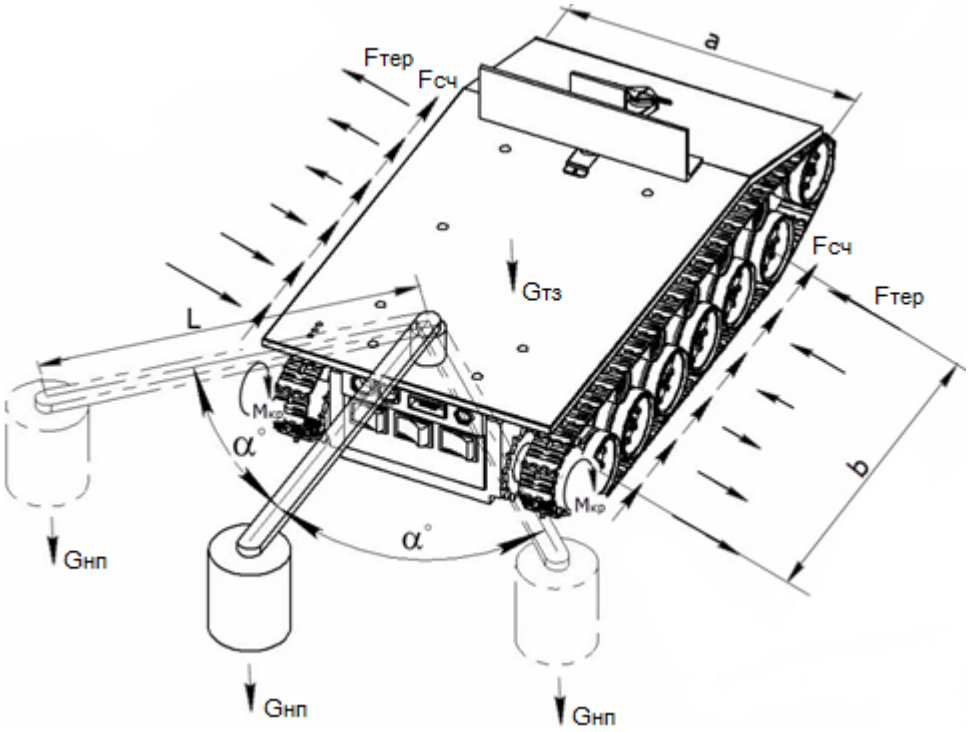

Рис. 1.3. Схема для розрахунку параметрів машини з гусеничним рушієм

Як бачимо, обґрунтування параметрів і створення системи керування гусеничним транспортним засобом з навісним обладнанням на основі аналізу впливу його конструктивних характеристик на величину кута повороту цього засобу є **актуальним науковим завданням**.

Автор поставив собі за **мету** розробити рекомендації до проектування системи керування гусеничним транспортним засобом, що обладнаний навісним пристроєм.

Досягнення поставленої в роботі мети має бути забезпечено шляхом:

1. Розробки комп'ютерної моделі гусеничного транспортного засобу.

2. Проведення на її базі обчислювального експерименту й аналізу отриманих результатів.

3. Створення системи керування.

4. Виконання фізичного експерименту на реальній моделі.

РОЗДІЛ 2
РОЗРОБКА Й ДОСЛІДЖЕННЯ КОМП'ЮТЕРНОЇ МОДЕЛІ ГУСЕНИЧНОГО ТРАНСПОРТНОГО ЗАСОБУ

2.1. Створення тривимірної комп'ютерної моделі гусеничного транспортного засобу

Вихідним технічним об'єктом для нашої подальшої роботи нам слугує діюча модель машини з гусеничним рушієм (рис. 2.1). На рисунку маємо такі позначення: 1 – привідна зірочка; 2 – напрямні колеса, 3 – опорні котки, 4 – підтримувальні котки, 5 – гусениця. Приводи правої та лівої гусениць сконструйовано як незалежні один від одного.

Рис. 2.1. Діюча модель машини з гусеничним рушієм

Як зазначалось вище, статику, кінематику й динаміку процесів у гусеничному рушії можна описати у вигляді складних нелінійних моделей. Таким чином, для дослідження механічного руху підходить інструмент програмного забезпечення SOLIDWORKS Motion, саме він дає можливість розв’язувати задачі на розрахунок механічного руху об’єктів.

Беручи за прототип модель танка (рис. 2.1), ми створили комп'ютерну модель транспортного засобу з навісним пристроєм (рис. 2.2). Після проведеного аналізу процесу складання комп'ютерної моделі не було виявлено конфліктів між деталями та вузлами машини, модель визнано працездатною.

Засобами програми SOLIDWORKS було обчислено масово-інерційні характеристики моделі, результати розрахунків подано на рис. 2.3.

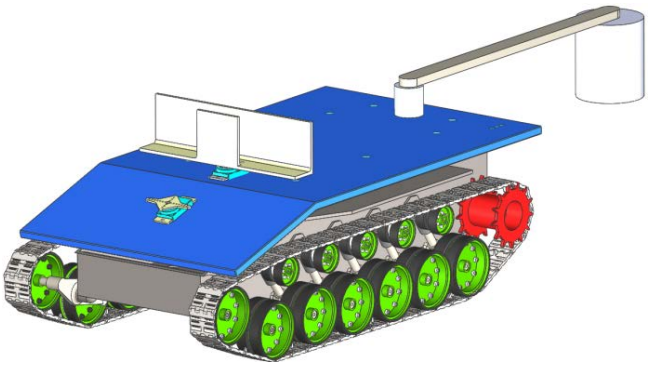

Рис. 2.2. Твердотіла комп'ютерна модель транспортного засобу з навісним пристроєм

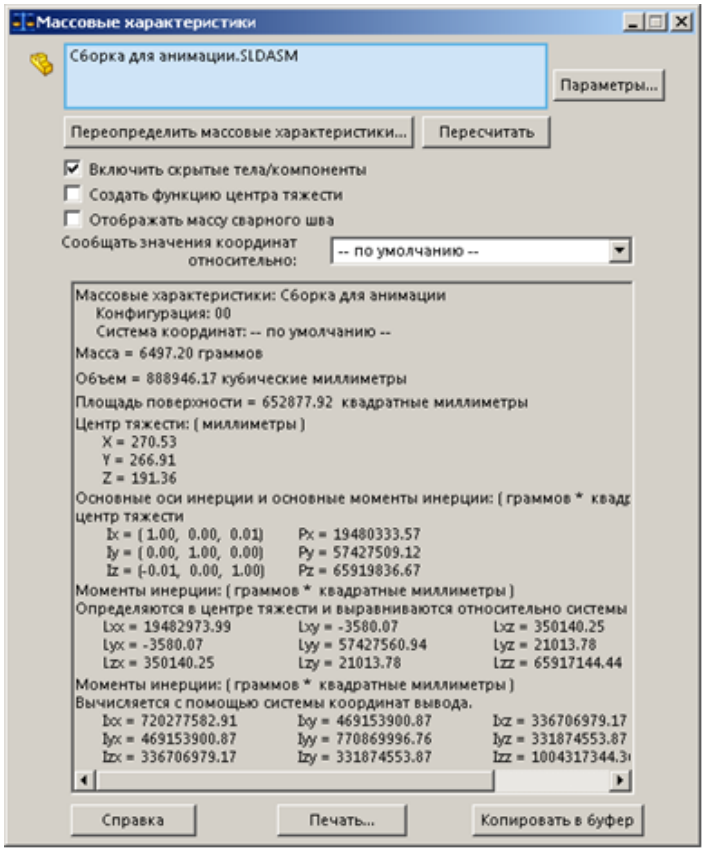

Рис. 2.3. Масово-інерційні характеристики твердотілої моделі гусеничного ТЗ

Щоб скоротити обсяг завдання, спростимо розрахункову модель. Зробимо припущення про однакову роботу всіх опорних котків. Побудуємо за допомогою інструмента SOLIDWORKS спрощену модель транспортного засобу з навісним пристроєм (рис. 2.4), що має масово-інерційні характеристики, близькі до параметрів твердотілої моделі. Назвемо наш спрощений варіант розрахунковою моделлю транспортного засобу з навісним пристроєм.

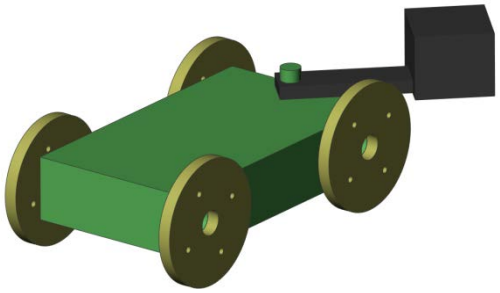

Рис. 2.4. Розрахункова модель транспортного засобу з НП

Для розрахункової моделі (рис. 2.4) було обрано такі значення параметрів, щоб її маса та моменти інерції (рис. 2.5) відрізнялися не більше як на 5 % від тих самих показників твердотілої моделі (рис. 2.3).

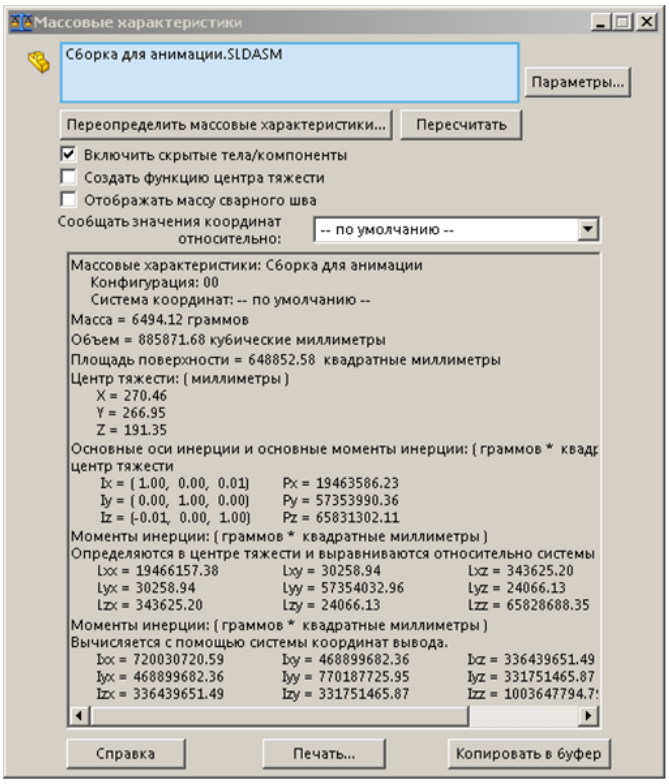

Рис. 2.5. Масово-інерційні характеристики розрахункової моделі гусеничного ТЗ

Для визначення масово-інерційних характеристик навісного пристрою було введено систему координат, пов'язану з віссю його повороту (рис. 2.6, *а*).

Отже, ми побудували розрахункову модель для проведення обчислювального експерименту з метою визначення кута повороту ТЗ в умовах, коли відхилення навісного пристрою, відбувається під різним кутом.

2.2. Визначення характеристик комп'ютерної моделі гусеничного транспортного засобу

Проведений за допомогою програми SOLIDWORKS Motion обчислювальний експеримент, протягом якого було визначено кут повороту ТЗ коли навісний пристрій відхилявся під різним кутом, базувався на умовах, що показані на рис. 2.5, 2.6.

а *б*

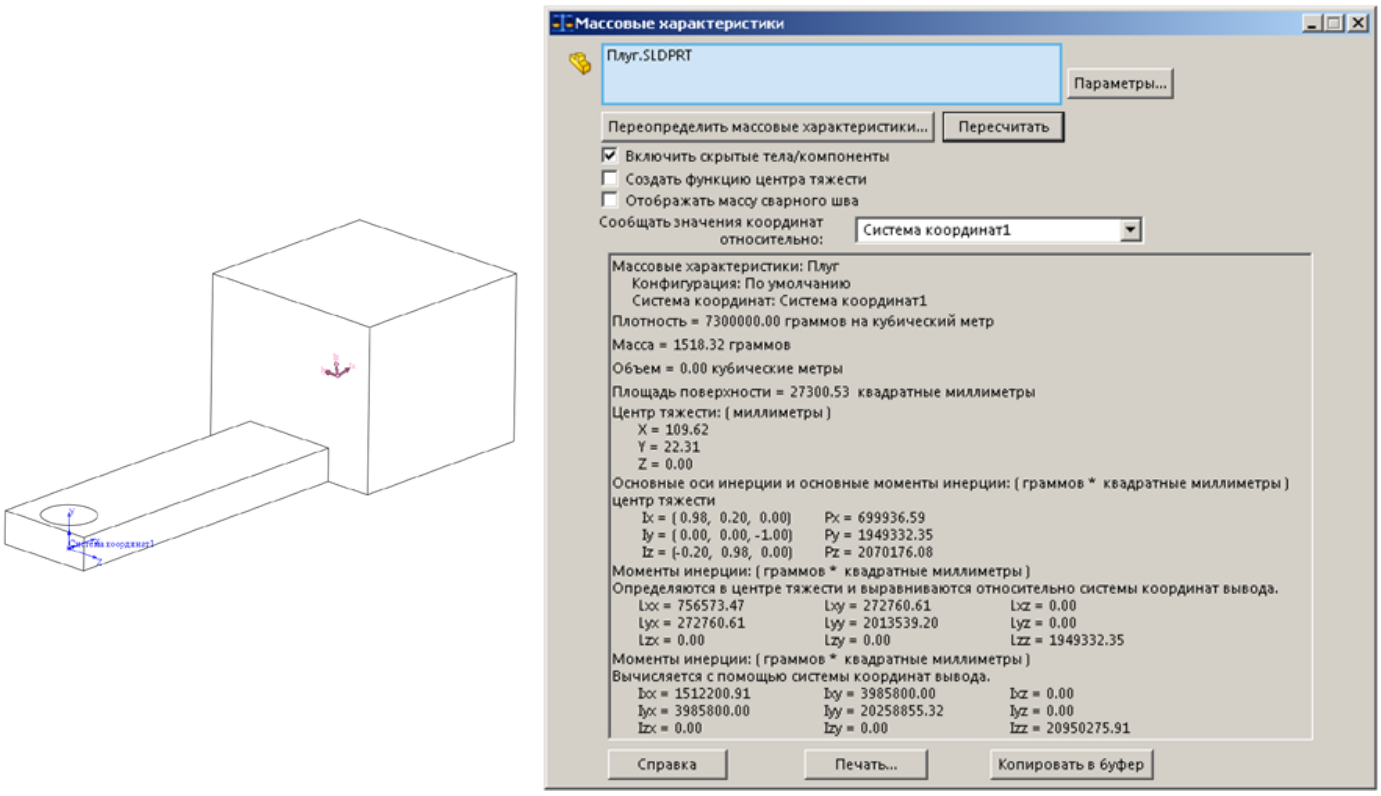

Рис. 2.6. Умови для дослідження моделі НП: *а* – схема каркаса моделі з новою системою координат; *б* – масово-інерційні характеристики

Вихідні дані та результати обчислювального експерименту наведено в табл. А.1, А.2 додатка.

У побудові моделей було використано такі інструменти програми SOLIDWORKS Motion: крутний момент, лінійний двигун і сила тяжіння (див. рис. 2.7). До того ж на рисунку побудовано стосовно однієї з моделей траєкторії руху початкових і кінцевих точок кожної гусениці, а також відображено дерево конструювання та панель інструментів SOLIDWORKS Motion.

Розглянемо, як відбувались обчислення параметрів стосовно кожної з виділених нами схем бокового повороту ТЗ.

Експеримент 1. «Плавний поворот». Для виконання обчислень скористаємось зображеними на рис. 2.8 траєкторіями руху гусениць ТЗ, кути повороту якого відповідають таким значенням кута повороту НП: 0, 45, -45 градусів.

Відповідно до цих умов уведемо такі позначення: $\mu = \frac{m_{НП}}{m_{ТЗ}}$, $\beta_1 = \frac{\alpha_{45^\circ}}{\alpha_{0^\circ}}$, $\beta_2 = \frac{\alpha_{-45^\circ}}{\alpha_{0^\circ}}$.

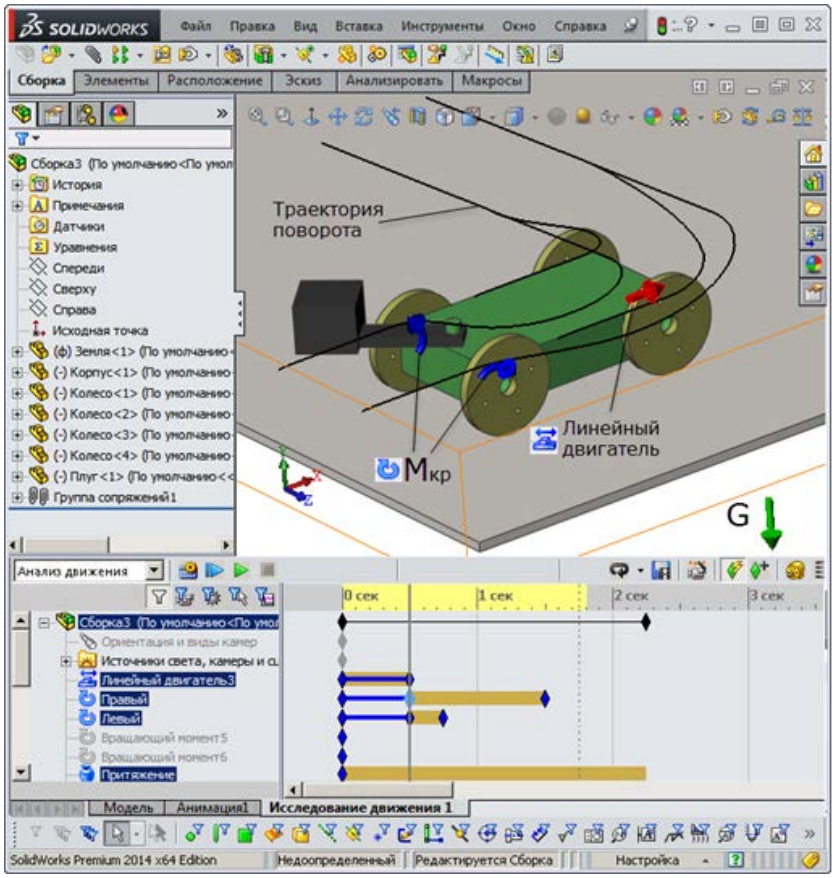

Рис. 2.7. Схема проведення обчислювального експерименту

За даними трьох варіантів експерименту (див., табл. А.1 додатка) на рис. 2.9 побудовано криві залежності безрозмірних кутів повороту ТЗ від безрозмірної маси НП. Ці залежності свідчать, що при збільшенні відносної маси навісного пристрою ефективність повороту підвищується.

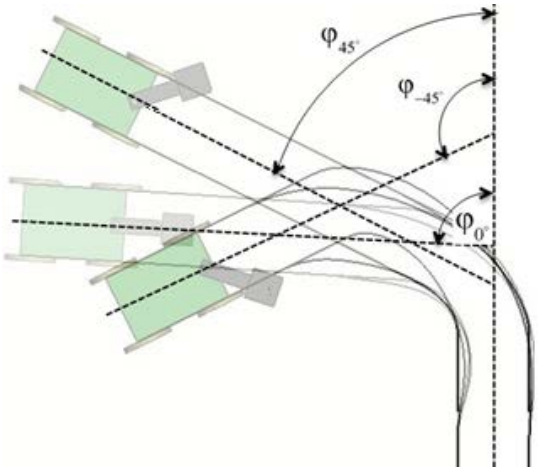

Рис. 2.8. Траєкторії руху гусениць ТЗ при його плавному повороті

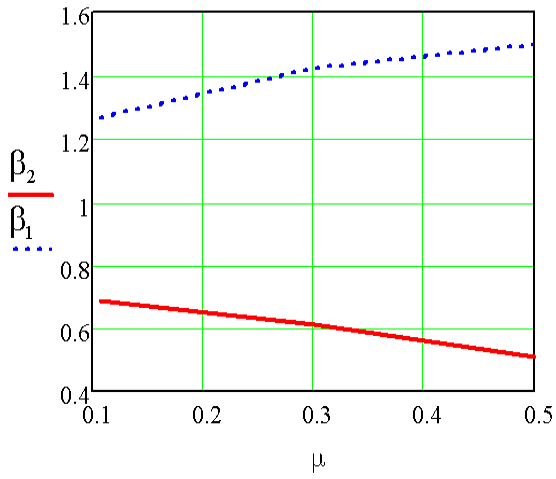

Рис. 2.9. Графіки залежності безрозмірних кутів повороту ТЗ від безрозмірної маси НП

Експеримент 2. «Крутий поворот». Для цього випадку розглянемо побудовані на рис. 2.10 траєкторії руху гусениць ТЗ, кути його відповідають значенням кутів повороту НП, що становлять 0, 45, –45 градусів.

Скориставшись результатами трьох варіантів експерименту (див. табл. А.2 додатка) будуємо криві залежності безрозмірних кутів повороту ТЗ від безрозмірної маси НП (див. рис. 2.11). Як бачимо, збільшення відносної маси навісного пристрою зумовлює зниження ефективності повороту ТЗ.

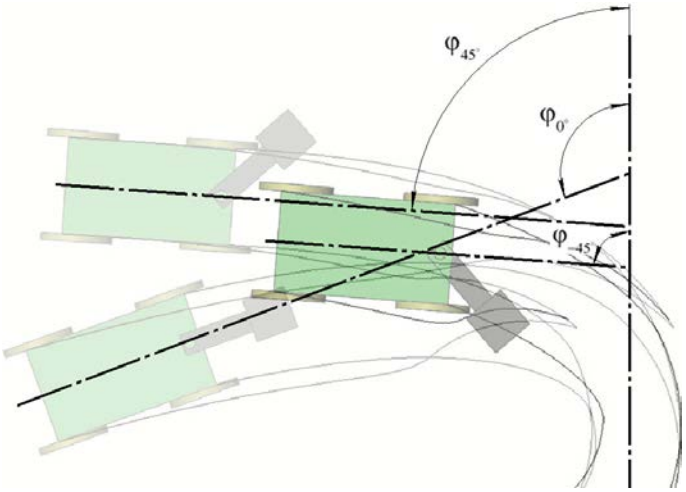

Рис. 2.10. Траєкторії руху гусениць ТЗ при крутому повороті

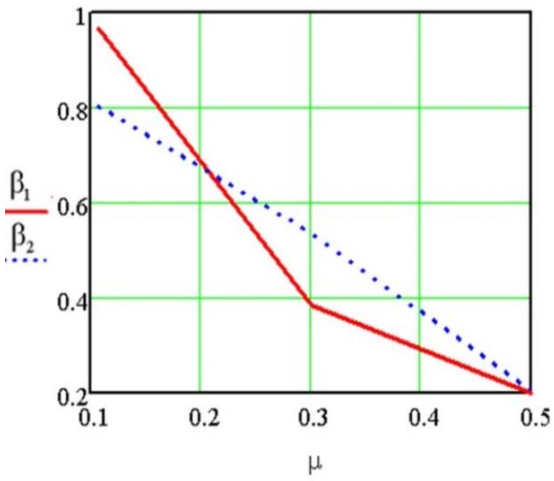

Рис. 2.11. Графіки залежності безрозмірних кутів повороту ТЗ від безрозмірної маси НП

Експеримент 3. «Поворот на місці». Моделювання даної схеми руху дозволило визначити траєкторії руху гусениць ТЗ (рис. 2.12), кути повороту якого відповідають таким значенням кута повороту НП: 0, 45, –45 градусів.

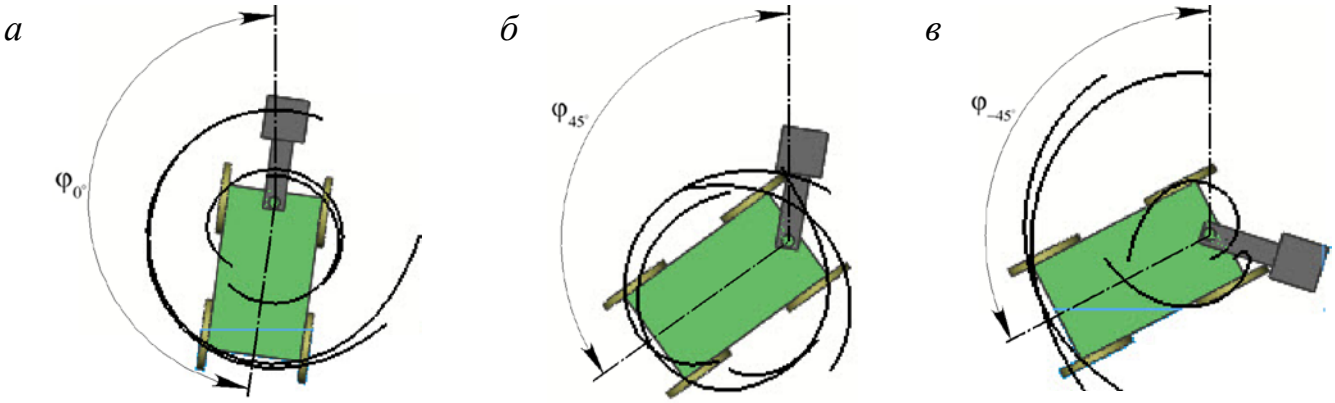

Рис. 2.12. Траєкторії руху гусениць при повороті ТЗ на місці

На базі результатів трьох варіантів експерименту (див. табл. А.3 додатка) було побудовано залежність безрозмірних кутів повороту ТЗ від безрозмірної маси НП (рис. 2.13). Вони показали, що збільшення відносної маси навісного пристрою веде до зниження ефективності повороту.

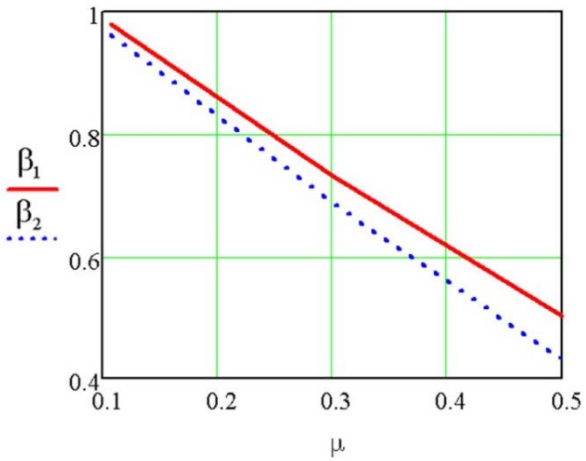

Рис. 2.13. Графіки залежності безрозмірних кутів повороту ТЗ від безрозмірної маси НП

2.3. Висновки

Після проведення обчислювальних експериментів на базі комп'ютерної моделі гусеничного транспортного засобу вперше доведено такі залежності:

1. Коли гусеничний ТЗ рухається за схемою «плавний поворот», то значення кута його повороту лінійно залежить від маси навісного пристрою, причому ефективність повороту підвищується при відхиленні цього пристрою в бік центра кривизни траєкторії машини.

2. Рух гусеничного транспортного засобу відповідно до схем «крутий поворот» і «поворот на місці» зумовлює досягнення оптимального кута повороту, коли навісний пристрій займає положення на поздовжній осі машини.

3. На траєкторію руху транспортного засобу з навісним пристроєм виявляють вплив такі фактори: схема бортового повороту, кут відхилення навісного пристрою від поздовжньої осі транспортного засобу, а також відносна маса навісного пристрою.

РОЗДІЛ 3
ВИГОТОВЛЕННЯ МАКЕТУ Й СТВОРЕННЯ СИСТЕМИ КЕРУВАННЯ

3.1. Розробка системи керування транспортним засобом

Як показав розгляд вихідного технічного об'єкта, керування ним здійснюється з пульта, а це не дозволяє задати необхідну програму руху машини, а значить, ефективно реалізувати потрібний режим її повороту, у зв’язку з чим спостерігається перегрівання транзисторів на драйвері двигунів, а також недостатня потужність акумуляторів.

Використовуючи результати експериментів, описаних у попередньому розділі, автор поставив завдання розробки системи керування гусеничним транспортним засобом з навісним пристроєм. При цьому було вирішено в ролі пульта керування моделлю використати комп'ютер. Для налагодження зв'язку між комп'ютером і моделлю застосували проміжний пристрій у вигляді смартфона, що функціонує в операційній системі Android.

З метою скорочення обсягу цієї роботи, обґрунтування та алгоритм розробки схеми керування винесено в додаток Б. У написанні програмної частини процес розробки системи використовували мови програмування HTML, CSS, JavaScript, Python, C. Для створення й налагодження програмного забезпечення роботи виконавця було використано інтегроване середовище створення програм для мікроконтролерів фірми Microchip MPLAB v.9; блок-схему програмної частини системи керування на рис. 3.1 (детальніше – у додатку Б). Програмний код, що запускається на веб-сервері, написано мовами HTML, Python, CSS, JS (див. додаток В), а програмний код виконавця – мовою C (див. додаток Г).

3.2. Виготовлення макету системи керування робота

Загальний вигляд електронної частини системи керування, розробленої автором, зображена на рис. 3.2. У неї входять такі елементи: 1 – макетна плата, на якій розміщено мікроконтролер PIC16F877A, 2 – плата драйвера двигунів L298, 3 – плата-компаратор, 4 – два блоки акумуляторів, 5 – плата перемикання напруги, 6 – плата датчика TSOP.

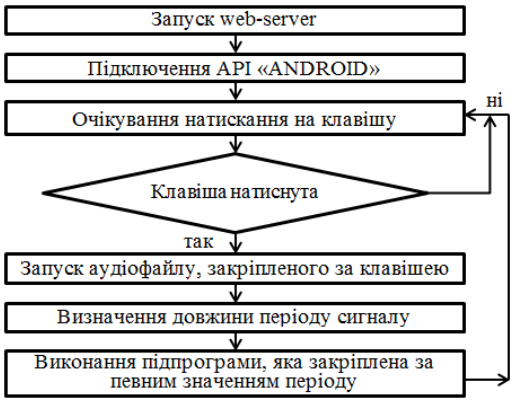

Рис. 3.1. Блок-схема програмної частини системи керування ТЗ

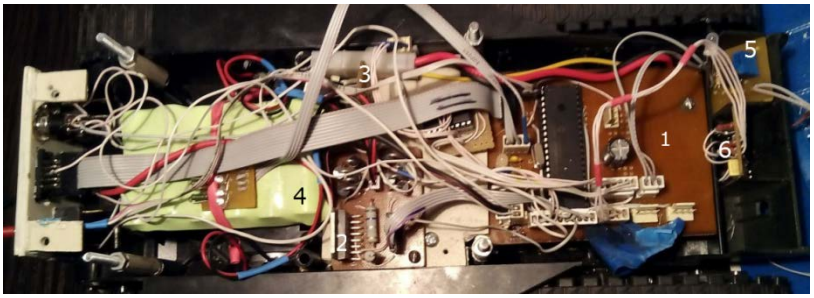

Рис. 3.2. Загальний вигляд системи керування транспортним засобом

Верхня кришка (рис. 3.3) слугує для прикриття електронної частини робота, на ній розміщено датчик TSOP (1); утримувач для телефона (2).

Рис. 3.15. Верхня кришка робота

Систему керування було випробувано, що підтвердило її працездатність.

3.3. Висновки

1. З використанням результатів експериментів, описаних у розділі 2, було розроблено систему керування гусеничним транспортним засобом, що обладнаний навісним пристроєм. За допомогою системи вдалось задати необхідну програму руху машини.

2. Схему керування побудовано за принципом постійного умовного зв’язку між веб-сервером і системою. До схеми входять пульт керування – комп'ютер, смартфон і робот. Для підтримки функцій системи керування розроблено необхідне програмне забезпечення.

3. Тестування системи керування показало її високу надійність у роботі.

РОЗДІЛ 4
ВИПРОБУВАННЯ МОДЕЛІ ГУСЕНИЧНОГО ТРАНСПОРТНОГО ЗАСОБУ

За допомогою натурної моделі було досліджено крутий поворот ТЗ, що відповідає різним значенням кута відхилення навісного пристрою. На рис. 4.1 зображено траєкторії руху ТЗ, що утворились за різних умив експерименту (НП розташований на поздовжньої осі ТЗ; НП відхилений назовні від центра повороту ТЗ; НП відхилений у бік центра повороту ТЗ). Порівняння результатів натурного та обчислювального експериментів, що стосувались режиму руху ТЗ для схеми «крутий поворот», зведено в табл. 4.1.

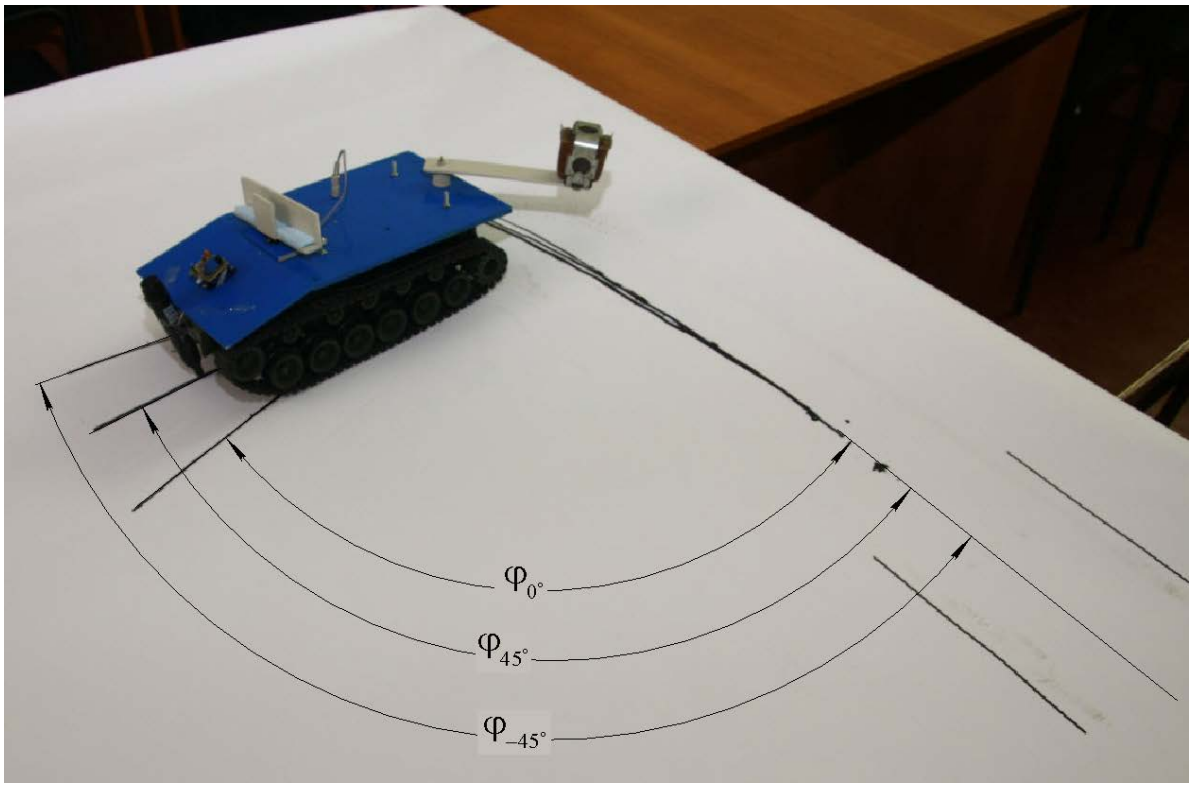

Рис. 4.1. Результат проведення випробувань натурної моделі ТЗ

Дані табл. 4.1 показують, що різниця між результатами комп'ютерного і натурного експериментів не перевищує 17 %.

Тим самим підтверджено адекватність розрахункової моделі SOLIDWORKS Motion та натурної моделі, похибка якої не перевищує 17 %.

Таблиця 4.1

Результати порівняння натурного випробування й обчислювального експерименту стосовно крутого повороту ТЗ

Кут повороту НП α, град	Кут повороту ТЗ φ, град, при дослідженні	
	натурної моделі	комп'ютерної моделі
0	92	106
45 (від центра кривизни)	89	104
-45 (до центра кривизни)	74	89

Висновки

У результаті дослідження режиму «крутий поворот» за допомогою діючої фізичної моделі гусеничного транспортного засобу з навісним пристроєм підтверджено її адекватність розрахунковій моделі того самого транспортного засобу (рис. 2.4). Похибка моделювання процесу засобами SOLIDWORKS Motion не перевищує 17 %.

ВИСНОВКИ

Науково-дослідна робота являє собою закінчену наукову працю, у якій виконано актуальне наукове завдання, що полягає в обґрунтуванні параметрів та у створенні системи керування гусеничним транспортним засобом з навісним пристроєм.

Основні наукові результати, висновки та рекомендації:

1. Гусеничний рушій і система керування ним – це той компонент машини, що суттєво впливає на її тягові властивості, тривалість експлуатації, економічність і надійність. Найчастіше такі машини мають навісне обладнання, схильне до зміщення центру ваги, а це ускладнює процес руху й маневрування засобів.

2. Технічною ідеєю роботи було визнано підвищення ефективності керування машиною з гусеничним рушієм за рахунок повороту навісного пристрою.

3. Було встановлено, що механіку процесів у гусеничному рушії можна описати як складні нелінійні моделі. З огляду на це, дослідження механічного руху вирішили здійснювати, задіявши інструмент програми SolidWorks Motion.

4. Виявилось, що коли гусеничний транспортний засіб рухається за схемою «плавний поворот», значення його кута повороту лінійно залежить від маси навісного пристрою, причому ефективність процесу підвищується при відхиленні навісного пристрою в бік центра кривизни траєкторії руху ТЗ.

5. Рух гусеничного транспортного засобу за схемами «крутий поворот» і «поворот на місці» характеризується таким чином, що оптимальне значення кута повороту можливе в момент розташування навісного пристрою на поздовжній осі машини.

6. Проведення натурного експерименту дозволило підтвердити адекватність розрахункової моделі SOLIDWORKS Motion натурній моделі ТЗ, причому похибка результатів не перевищила 17 %.

7. Було доведено експериментально практичну цінність розробленої автором системи керування, оскільки вона забезпечує мінімальний час зв’язку між пристроєм і веб-сервером, дає можливість виконувати поставлені завдання за алгоритмом виконавця, транслювати зображення на екран пульта керування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Платонов В.Ф. Динамика и надежность гусеничного движителя / В.Ф. Платонов. – М. : Машиностроение, 1973. – 232 с.

2. Антонов А.С. Теория гусеничного движителя / А.С. Антонов. – М. : Машгиз, 1949. – 354 с.

3. Позин Б.М. Задачи пассивного поворота гусеничной машины / Б.М. Позин, И.П. Трояновская // Вестник Южно-Уральского государственного университета. Серия: Машиностроение. – 2007. – Вып. № 25 (97). – С. 70–74.

Додаток А

ВИХІДНІ ДАНІ Й РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ ХАРАКТЕРИСТИК КОМП'ЮТЕРНОЇ МОДЕЛІ ГУСЕНИЧНОГО ТРАНСПОРТНОГО ЗАСОБУ

Таблиця А.1

Вихідні дані обчислювального експерименту і результати при плавному повороті ТС

Характеристики	№ експерименту		
	а	б	в
Радіус колеса	100	100	100
Поперечна база – a, мм	137	137	137
Поздовжня база – b, мм	195	195	195
Довжина стріли – l, мм	80	80	80
Маса ТЗ – m_{TC} , кг	14,06	14,06	14,06
Масса НП – $m_{HУ}$, кг	1,52	4,22	7,03
Кут повороту НП – α, град:	Кут повороту ТЗ – φ, град:		
0	90	90	90
45 (від центру кривизни)	62	55	46
-45 (до центру кривизни)	114	128	135

Таблиця А.2

Вихідні дані обчислювального експерименту і результати при крутому повороті ТЗ

Характеристики	№ експерименту		
	а	б	в
Радіус колеса	100	100	100
Поперечна база – a, мм	137	137	137
Поздовжня база – b, мм	195	195	195
Довжина стріли – l, мм	80	80	80
Маса ТЗ – m_{TC} , кг	14,06	14,06	14,06
Масса НП – $m_{HУ}$, кг	1,52	4,22	7,03
Кут повороту НП – α, град:	Кут повороту ТЗ – φ, град:		
0	92	93	118,5
45 (від центру кривизни)	89	36	23,8
-45 (до центру кривизни)	74	50	24,2

Таблиця А.3

Вихідні дані обчислювального експерименту і результати для повороту ТЗ на місці

Характеристики	№ експерименту		
	а	б	в
Радіус колеса	100	100	100
Поперечна база – a, мм	137	137	137
Поздовжня база – b, мм	195	195	195
Довжина стріли – l, мм	80	80	80
Маса ТЗ – m_{TC} , кг	14,06	14,06	14,06
Масса НП – $m_{HУ}$, кг	1,52	4,22	7,03
Кут повороту НП – α, град:	Кут повороту ТЗ – φ, град:		
0	315	172	118,5
45 (від центру кривизни)	309	125,5	59,5
-45 (до центру кривизни)	303	118	51

Додаток Б

СТВОРЕННЯ СИСТЕМИ КЕРУВАННЯ ГУСЕНИЧНИМ ТРАНСПОРТНИМ ЗАСОБОМ

Б.1. Розробка функціональної схеми керування ТЗ

Схема управління побудована за принципом постійного умовного спілкування між веб-сервером і роботом, за допомогою аудіо роз'єми (audio-jack).

На смартфоні запускається web-server зі скриптом, який написано на мові Python. У цьому файлі реалізується відображення веб сторінки у вікні браузера, обробка натискання кнопок, передача даних з камери, відтворення аудіо файлів і підключення до API смартфона. Блок схема програми системи управління показана на (рис. Б.1).

Рис. Б.1

Блок схема функціонального управління

Запуск скрипта відбувається на смартфоні в середовищі SL4A. При відкритті даної програми відкривається вікно, в якому і потрібно вибрати скрипт, який буде запущений (рис. Б.2).

Рис. Б.2

Скріншот програми SL4A, запуск скрипта

Scripting Layer for Android одночасно дозволяє запуск скрипта і в теж час редагування його не посередньо на самому телефоні. Коли натискається на пульті управління кнопка або клавіша управління руху роботом, то на смартфон, а конкретно на web-server йде сигнал через бездротове підключення WIFI. При такій маніпуляції в Scripting Layer for Android виникає індикація, що натиснута клавіша (рис. Б.3).

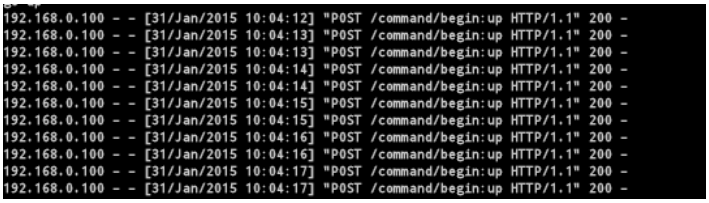

Рис. Б.3

Скріншот програми SL4A, відповідь від сервера

На підставі цієї індикації можна сказати, що відправка пакету даних йде кожні 200 мс. За допомогою аудіо роз'єму смартфон з'єднаний з моделлю, а саме компаратором, який робить з синусоїдного аудіо файлу, квадратичну хвилю. Вже готова квадратична хвиля піднімається до потрібного рівня сигналу, щоб мікроконтролер міг розпізнавати значення 1/0. Сигнал потрапляє на ніжку мікроконтролера, де і відбувається визначення тривалості періоду. Так як частоти відносно великі, починаючи від 100Hz, то за рахунку потрібна велика точність. Як тільки було визначено значення тривалість періоду, мікроконтролер обробляє програму, яка призначена для отриманого числа. Після виконання цього циклу мікроконтролер чекає наступних команд. Після приходу нової хвилі цикл повторюється.

Б.2. Створення електричної принципової схеми керування з обґрунтуванням елементної бази

Серцем пристрою можна називати мікроконтролер PIC16F877A фірми MICROCHIP. Peripheral Interface Controller, що означає «контролер інтерфейсу периферії». У номенклатурі Microchip Technology Inc. представлений широкий спектр 8-й, 16-й і 32-бітних мікроконтролерів і цифрових сигнальних контролерів під маркою PIC. Відмінною особливістю PIC-контролерів є хороша спадкоємність різних сімейств. Це і програмна сумісність (єдина безкоштовна середовище розробки MPLAB IDE), і сумісність за висновками, по периферії, по напруженням живлення, за коштами розробки, по бібліотеках і стеком найбільш популярних комунікаційних протоколів. Номенклатура налічує більше 500 різних контролерів з усілякими варіаціями периферії, пам'яті, кількістю висновків, продуктивністю, діапазонами живлення і температури і т. п.

8-бітові мікроконтролери PIC10 / 12/16 представлені двома базовими архитектурами ядра: BASELINE і MID-RANGE.

Базова архітектура (BASELINE) складається з контролерів сімейства PIC10 і частини контролерів сімейств PIC12 і PIC16. Ґрунтуються вони на 12-й розрядної архітектури слова програм і представлені контролерами в корпусах від 6 до 28-і висновків. Спрощена архітектура базового сімейства надає найбільш дешеве рішення з пропонованих Microchip.

Мікросхема, яка відповідає за пересування робота в просторі це драйвер для управління електродвигунами невеликої потужності (чотири незалежні канали, об'єднаних у дві пари). Крім того, у L293D є два входи для включення кожного з драйверів. Ці входи використовуються для керування швидкістю обертання електромоторів за допомогою широтно модульованого сигналу (ШІМ).

L293D забезпечує поділ електроживлення для мікросхеми та для керованих нею двигунів, що дозволяє підключити електродвигуни з великою напругою живлення, ніж у мікросхеми. Поділ електроживлення мікросхем і електродвигунів

може бути також необхідно для зменшення перешкод, викликаних кидками напруги, пов'язаними з роботою моторів.

Але в ході експериментів і розрахунків цей дайвер двигуна був замінений на більш потужний. Потужність L298 була більше в 2 рази.

Завдяки наявності двох мостів мікросхема може керувати двома двигунами постійного струму, причому незалежно (рис. Б.4).

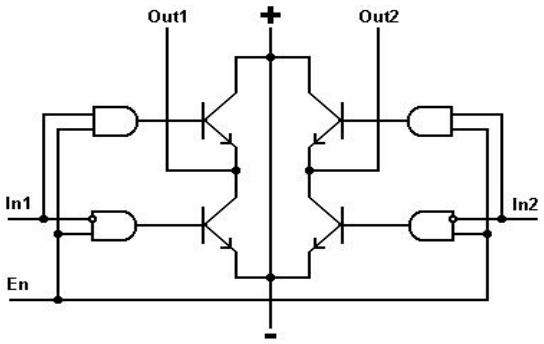

Рис. Б.4

Блок схема двох мостів L298N

Схема керування двигунами постійного струму взята з офіційного джерела (datasheet) (рис. Б.5). Характеристики схем представлені в (табл. Б.1).

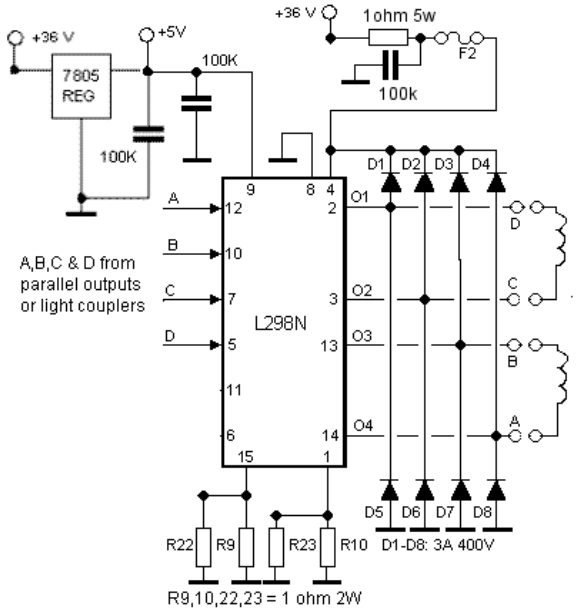

Рис. Б.5

Електрична принципова схема драйвера L298

Таблиця Б.1.

Характеристики драйвера L298

Напруга живлення двигунів	до 50 V
Напруга живлення мікросхеми	7 V
Максимальний піковий струм ключів (t <100 мксек)	3 А
Середній (сталий) струм ключів	2 А
Споживаний мікросхемою струм не більше	70 мА
Потужність що розсіюється	25 Ватт
Вхідний рівень «Лог 0» менш	1,5 V
Вхідний рівень «Лог 1» більш	2,3 V
Падіння напруги на ключах при струмі 1 А	не більше 1,7 В
Падіння напруги на ключах при струмі 2 А	не більше 2,7 В

Для керування швидкості пересування робота прийнято рішення зробити «перемикач» напруги живлення двигунів. У цій системі перемикання живлення використовується польові транзистори IRF9630 (MOSFET) (рис. Б.6).

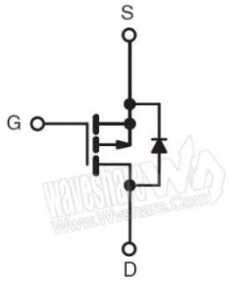

Рис. Б.6

Електрична принципова схема IRF9630

Розроблено плату, на якій розміщується 2 польових транзистора, 4 резистора номіналом 1к. і ключові транзистори КТ315Г. Вийшло, що перша напруга дорівнює повній батареї, тобто 12 В, а друга напруга становить 8,4 В.

Компаратор зроблено по електричній принциповій схемі, яка опублікована в інтернеті (рис. Б.7).

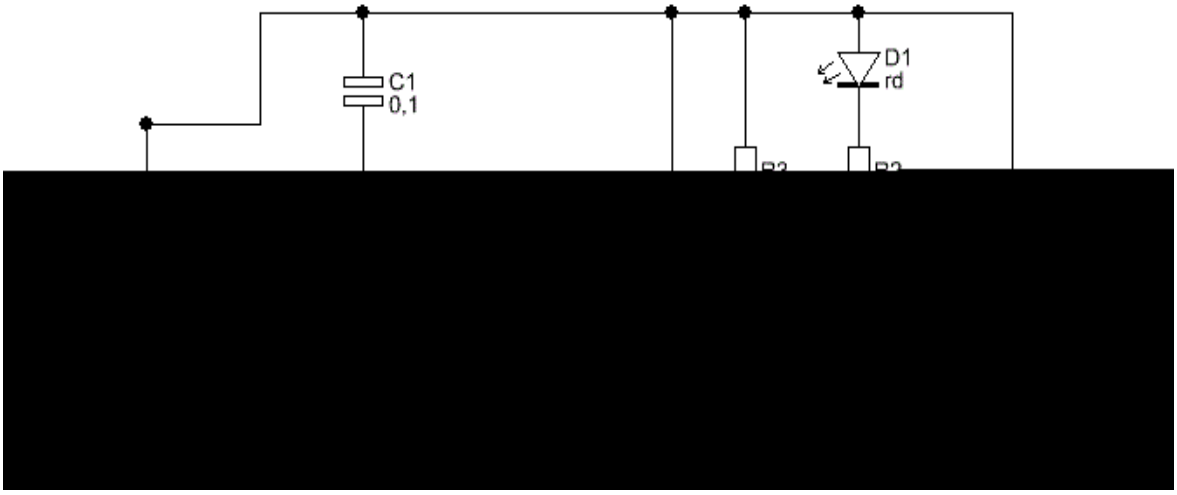

Рис. Б.7

Електрична принципова схема компаратора

Для створення аудіо файлу використовувалася програма Audacity (рис. Б.8). У ній була створена синусоїда 1000Гц тривалістю 30 секунд і збережена в WAV файл. Прийнято рішення зробити тестовий файл з частотою 10Гц. Після відтворення цієї синусоїди з частотою 10Гц світлодіод блимав стабільно і не було ніяких провалів. Звичайно, прийнятий сигнал після компаратора виявляється «Оцифрованим», є тільки нуль і одиниця - це сигнал з прямокутними імпульсами. Тим не менш, частота «оцифрованого» сигналу дорівнює частоті, що подається.

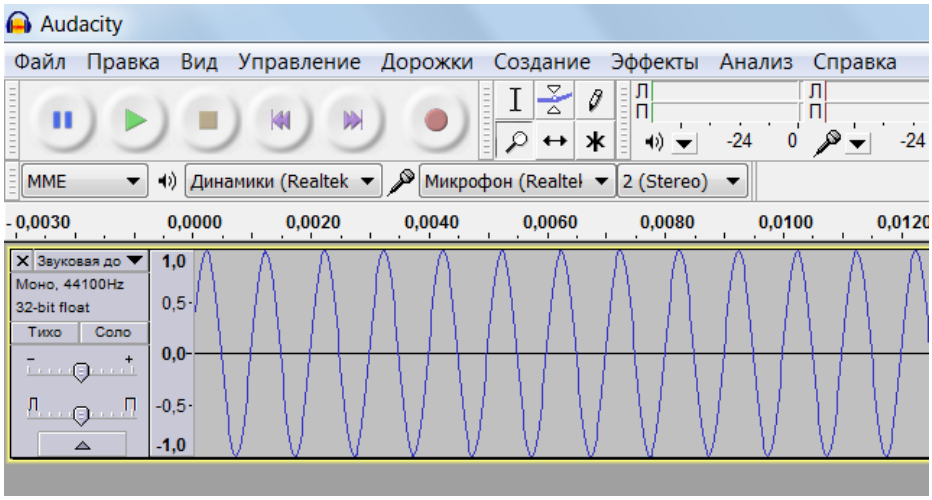

Рис. Б.8

Скріншот програми Audacity, синусоїда

Для виявлення перешкод використана технологію інфрачервоного випромінювання та інфрачервоного приймача. В якості ІК випромінювача в досліджуваному роботі використовується ІЧ світлодіод, а в якості ІЧ приймача використовується TSOP який працює на частоті 36khz. Що б TSOP відбитий промінь світла від перешкоди то ІК випромінювач повинен працювати на одній і тій же частоті що і TSOP, а саме 36khz.

Робота цього модуля. За допомогою ШИМа на ніжці мікроконтролера створена частота 36khz. До С1 і С2 підключені катоди ІЧ світлодіодів, які знаходяться на невеликих модулях «материнській платі».

ШИМ – це управління середнім значенням напруги на навантаженні шляхом зміни шпаруватості імпульсів, керуючих ключем.

Програмний запуск ШИМа на мові «С» в середовищі програмування MPLAB представлений на (рис. Б.9).

```
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);
setup_timer_2(T2_DIV_BY_1,131,1);
setup_ccp2(CCP_PWM);
set_pwm2_duty(264);
setup_ccp1(CCP_PWM);
set_pwm1_duty(264);
```

Рис. Б.9

Запуск ШИМа

При роботі ІЧ світлодіода випромінюється промінь, який спрямований у ту сторону, в яку і спрямований сам світлодіод. Пристрій визначає предмети навколо себе на відстані 15 – 30 см. Цей модуль розміщено на серво приводі. Коли промінь відбивається від предмета, який заважає руху робота і знаходиться в площині перед собою на 180 градусів, то TSOP приймає його і йде на обробку переривання.

Переривання – це зміна значень на бітах B0 – B7. Якщо змінюється значення рівня сигналу на цих пінах то мікроконтролер переходить у функцію обробки переривання. Після виконання програми обробки переривання він повертається в те

місце, звідки пішов у переривання. При написанні програми було прийнято рішення написання її без тимчасового опитування датчиків, тому що це уповільнює роботу реакції на виявлення перешкоди і не коректну роботу рахунки періоду. Перед використанням функції обробки переривань, потрібно дозволити самі переривання і прописати в програмі, коли вона буде викликатися. Для цього служить регістр (команда) INTCON.

Біт дозволу вказує на конкретний вид переривань. Прапор ж говорить про стан переривання (якщо він дорівнює 1, то переривання є, якщо 0, то переривання немає). Розглянемо, як і за що відповідає кожен біт регістра INTCON. Відраховуються біти справа на ліво, починаючи з нульового (рис. Б.10).

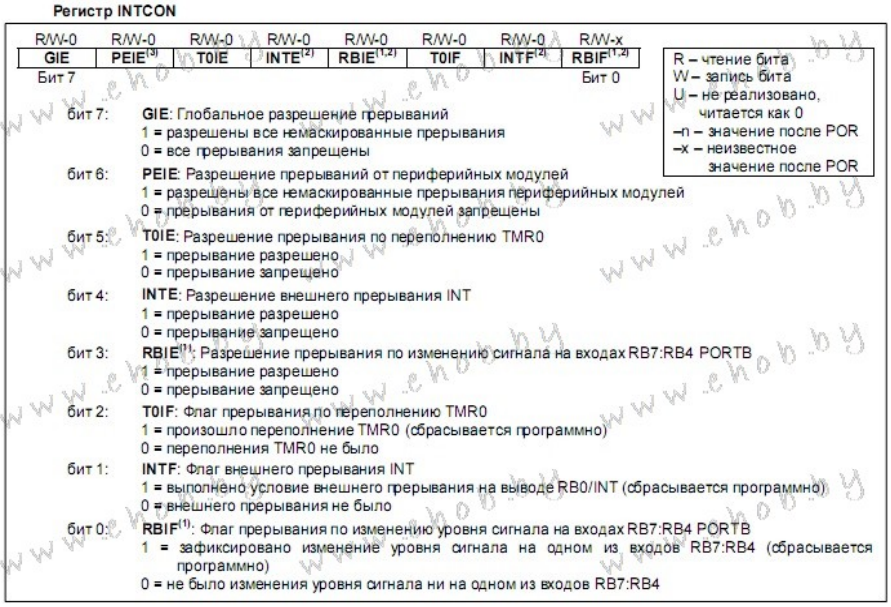

Рис. Б.10

Призначення бітів

Сервопривод (привід що стежить) (рис. Б.11) – привід з управлінням через негативний зворотний зв'язок, що дозволяє точно керувати параметрами руху. Сервоприводом є будь-який тип механічного приводу (пристрої, робочого органу), що має в складі датчик (положення, швидкості, зусилля і т. п.) і блок керування приводом (електронну схему або механічну систему тяг), що автоматично підтримує необхідні параметри на датчику (і, відповідно, на пристрої) згідно заданому

зовнішньому значенням (положенню ручки керування або чисельним значенням від інших систем).

Сервоприводи живляться від постійної напруги в межах від 4,8 В до 6 В. Зазвичайно використовується універсальний роз'єм, який містить три контакти: чорний – загальний провід або земля (GND), червоний – напруга живлення (+ Vcc), жовтий (або іншого кольору) – керуючий сигнал. Керуючий сигнал (рис. Б.12) – це імпульсний сигнал з ШІМ (широтно-імпульсною модуляцією), що представляє собою послідовність прямокутних імпульсів з амплітудою 3–5 В і тривалістю від 0,9 до 2,1 мс.

Рис. Б.12

Сигнали, що надходять на сервопривід

Номінальна періодичність проходження імпульсів зазвичай 50 Гц (інтервал – 20 мс), але сервоприводи зберігають працездатність і при досить сильному відхиленні цього параметра (15–20%). Таким чином, скважність керуючого сигналу дуже маленька – від 5% до 10%. Власне тривалість імпульсу і визначає положення виконавчого механізму. Мінімальне значення (1 мс) – означає розворот в крайнє ліве (або проти годинникової стрілки -40о ...- 80о залежно від моделі) становище, середнє значення (1,5 мс) – центральне положення штока, а максимальне значення (2 мс) – крайнє праве (за годинниковою стрілкою + 40о ... + 80о залежно від моделі) становище.

Б.3. Розробка програмної частини системи керування транспортним засобом

Блок-схему програмної частини системи керування на рис. Б.13 (детальніше – у додатку Б).

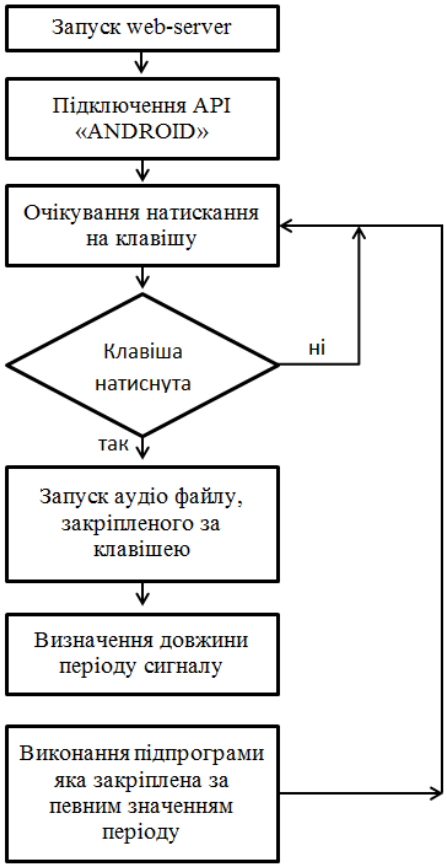

Рис. Б.13

Блок-схема програмної частини системи керування ТЗ

Scripting Layer For Android, веб-сервер (SL4A) виконує такі функції: запускає роботу веб-сервера (WebServer), який надсилає HTML сторінку, де і буде розміщено всю інформацію. У верхньому фреймі з’являється вікно трансляції з камери телефона. У нижньому фреймі – панель з кнопками типу start, stop, left, right.

Після натискання на кнопки html з панелі нижнього фрейма буде надсилатися запит HTTP-GET на телефон. Веб-сервер приймає та обробляє запити HTTP-GET і відповідно до них відтворює короткий звуковий ролик із сигналом змінної тональності. Ще одна важлива функція скрипту – ініціювати мовлення з камери

телефона. Мовлення ведеться на TCP порту 9091. Для цього було використано програму IP WebCam. Розподіл вікна браузера на два не залежні один від одного фрейми має забезпечити безперервність мовлення, поки діє запит HTTP-GET на керування машиною. Точку доступу створюють за допомогою стандартних можливостей смартфона, який працює в межах ОС Android. Смартфону автоматично присвоюється IP адрес (192.168.0.1).

Додаток В

ПРОГРАМНИЙ КОД ВЕБ-СЕРВЕРА

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# -*- codin by Baranik Vladisalv
ПОДКЛЮЧЕНИЕ БИБЛИОТЕК
import os
import BaseHTTPServer
from threading import Thread
import urlparse
import time
import string

try:
    import android
    droid = android.Android()
except ImportError:
    def mediaPlay(path):
        print "PLAY", path

    import mock
    droid = mock.Mock()
    droid.mediaPlay = mediaPlay



HOST_NAME = ''
PORT_NUMBER = 9090
COMMAND_SEND_INTERVAL = 0.5  # seconds
COMMAND_TIMEOUT = COMMAND_SEND_INTERVAL * 2
PAGE_TEMPLATE = string.Template('''
НАЧАЛО HTML СТРАНИЦИ
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
<title>The microprocessor control robotic systems</title>
<style type="text/css">
#action {
background:yellow;
border:0px solid #555;
color:#555;
width:0px;
```

```
height:0px;
padding:0px;
}
.btn{
color: #000;
display: inline-block;
margin: 0.35em 0.1em;
padding: 0.5em 1.2em;
outline: none;
text-decoration: none;
text-transform: uppercase;
letter-spacing: 1px;
font-weight: 700;
border-radius: 2px;

}
.btn:hover{
display: inline-block;
margin: 0.35em 0.1em;
padding: 0.5em 1.2em;
outline: none;
text-decoration: none;
text-transform: uppercase;
letter-spacing: 1px;
font-weight: 700;
border-radius: 2px;
border: 2px solid transparent;
}
.nazvanie {
position: absolute;
margin: 0;
padding: 0;
color: #f000;
text-align: center;
top: 5%;
left: 50%;
-webkit-transform: translate3d(-50%,-50%,0);
transform: translate3d(-50%,-50%,0);
}
table{
        width: 100%;
}
a {
```

```
        text-decoration: none;
        color: #bbb;
}
a:hover {
        text-decoration: underline;
        color: #000;
}
</style>
<script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"></script>

<script>
$(document).ready(function(){
    function CommandSender(cmdInterval){
        this.cmdInterval = cmdInterval;
        this.lastCmd = "";
        this.lastState = "";
        this.lastTime = new Date();
    }
    CommandSender.prototype.run = function(){
        var $this = this;
        setInterval(function(){
            $this.updateState($this.lastState, $this.lastCmd);
        }, this.cmdInterval);
    }
    CommandSender.prototype.updateState = function(state, cmd){
        if((new Date() - this.lastTime) < this.cmdInterval &&
            cmd == this.lastCmd && state == this.lastState)
            return;
        if (state == "begin" && cmd){
            this.sendCommand(state, cmd);
        }
        else if (state != this.lastState) {
            this.sendCommand("end", cmd);
        }
        this.lastState = state;
        this.lastCmd = cmd;
        this.lastTime = new Date();
    }
    CommandSender.prototype.sendCommand = function(state, cmd){
        $.post('command/' + state + ':' + cmd, function(resp){
            console.log(resp);
        });
```

```
    }

    var cmdInterval = ${COMMAND_SEND_INTERVAL};

    (function buttonsHandler(){
        var sender = new CommandSender(cmdInterval);
        $(".btn").mousedown(function(){
            sender.updateState("begin", $(this).data("command"));
        }).mouseup(function(){
            sender.updateState("end", $(this).data("command"));
        }).mouseleave(function(){
            sender.updateState("end", "");
        });
        sender.run();
    })();

    (function keyBoardHandler(){
        var sender = new CommandSender(cmdInterval);
        var arrow = {37:'left', 38:'up', 39:'right', 40:'down'};
        $(document).keydown(function (e) {
            var keyCode = e.keyCode || e.which;
            var cmd = arrow[keyCode];
            if (cmd)
                sender.updateState("begin", cmd);
        }).keyup(function (e) {
            sender.updateState("end", "");
        });
        sender.run();
    })();
});
</script>
</head>
<body>
        <h1 class="nazvanie">The microprocessor control robotic systems</h1>
<table style="margin-top: 30px;">
        <tr>
                <td>
                        <center style="margin-top: 100px">
                                <form name="myform" method="get">
                                        <button class="btn" data-
command="up">up</button>
                                        <button class="btn" data-
command="down">down</button>
```

```
                                        <button class="btn" data-
command="left">left</button>
                                        <button class="btn" data-
command="right">right</button>
                                    </form>
                                </center>
                                    <p style="margin-top: 70px">
                                        Робот начинает движение по нажатию на <text
style="font-weight: 700; font-size: 18px;">стрелки</text> (вверх, вних, влево,
вправо).
                                    </p>
                                    <p>
                                        Что бы переключить режим робота управляемого в
автономный вариант</br> нужно нажать на клавишу <text style="font-weight: 700; font-
size: 18px;">C</text> или на оборот.
                                    </p>
                                    <p>
                                        Что бы повернуть камеру нужно жать клавиши
<text style="font-weight: 700; font-size: 18px;">4</text> или <text style="font-
weight: 700; font-size: 18px;">6 </text>в зависимости от поворота в какую сторону.

                                    <center style="margin-top: 30%">автор роботы Бараник
Владислав</br><a href="https://vk.com/baranik.vladislav">Связь</a><a
href="http://baranik.com.ua" style="margin-left: 10px;">Сайт</a></center>
                                    </p>
                </td>
                <td>
                        <center><iframe src="http://${IP_ADDRESS}:8080/browserfs.html"
width="640" height="480" style="margin-top: 50px;"></iframe></center>
                </td>
        </tr>
</table>




</body>
</html>
''')


class CarControllerThread(Thread):
```

```
    path_to_audio = '/mnt/sdcard/robot'
    command_to_audio = {
        'up': '1_100s_hz.wav',
        'down': '1_200s_hz.wav',
        'left': '1_300s_hz.wav',
        'right': '1_400s_hz.wav',

    }

    def __init__(self, timeout=1):
        super(CarControllerThread, self).__init__()
        self.timeout = timeout
        self.pool_timeout = timeout / 3.0
        self.current_state = ""
        self.current_command = ""
        self.last_cmd_timestamp = time.time()

    def run(self):
        while True:
            elapsed = time.time() - self.last_cmd_timestamp
            if self.current_state == "begin" and elapsed > self.timeout:
                self.update_state("end", "")
            time.sleep(self.pool_timeout)

    def update_state(self, state, command):
        self.last_cmd_timestamp = time.time()
        if state == self.current_state and command == self.current_command:
            return
        self.current_state = state
        self.current_command = command
        if state == "end":
            print "stop"
            droid.mediaPlaySetLooping(False)
            droid.mediaPlayPause()
        elif state == "begin":
            print "go", command
            audio = self.command_to_audio.get(command)
            if audio:
                droid.mediaPlay(os.path.join(self.path_to_audio, audio))
                droid.mediaPlaySetLooping(True)

class DroidHandler(BaseHTTPServer.BaseHTTPRequestHandler):
    def do_HEAD(self):
```

```
        self.send_response(200)
        self.send_header("Content-type", "text/html; charset=utf-8")
        self.end_headers()

    def do_GET(self):
        self.send_response(200)
        self.send_header("Content-type", "text/html; charset=utf-8")
        self.end_headers()

        url = urlparse.urlsplit(self.path)

        if url.path == '/1.html':
            self.wfile.write(PAGE_TEMPLATE)
        else:
            ip, port = self.headers.get('Host').split(":", 2)
            self.wfile.write(
                PAGE_TEMPLATE.safe_substitute(
                    IP_ADDRESS=ip,
                    COMMAND_SEND_INTERVAL=int(COMMAND_SEND_INTERVAL * 1000),
                )
            )

    def do_POST(self):
        self.send_response(200)
        self.send_header("Content-type", "text/plain; charset=utf-8")
        self.end_headers()
        if self.path.startswith("/command/"):
            try:
                #/command/state:up
                state, cmd = self.path.split("/")[-1].split(":")
            except ValueError:
                print "ERROR", self.path
                self.wfile.write("fail")
                return
            controller.update_state(state, cmd)
            self.wfile.write("ok")
        elif self.path.startswith("/ping"):
            self.wfile.write("pong")


controller = CarControllerThread(COMMAND_TIMEOUT)
controller.start()
```

```
my_srv = BaseHTTPServer.HTTPServer((HOST_NAME, PORT_NUMBER), DroidHandler)
print 'web server running on port %s' % PORT_NUMBER
my_srv.serve_forever()
```

Додаток Г

ПРОГРАМНИЙ КОД ВИКОНАВЦЯ

```
#include "F:\Robots\Kiev\main.h"
long long count, i, r, l, s0, s180, notprer, schet, servo, program, nachalo;
void forward (void)
      {
      output_bit ( PIN_D2, 1);
      output_bit ( PIN_D3, 0);
      output_bit ( PIN_D4, 0);
      output_bit ( PIN_D5, 1);
      output_bit ( PIN_D6, 0);
      output_bit ( PIN_D7, 1);
   }
void back (void)
      {
      output_bit ( PIN_D2, 0);
      output_bit ( PIN_D3, 1);
      output_bit ( PIN_D4, 1);
      output_bit ( PIN_D5, 0);
      output_bit ( PIN_D6, 0);
      output_bit ( PIN_D7, 1);
   }
void left (void)
      {
      output_bit ( PIN_D2, 1);
      output_bit ( PIN_D3, 0);
      output_bit ( PIN_D4, 1);
      output_bit ( PIN_D5, 0);
      output_bit ( PIN_D6, 0);
      output_bit ( PIN_D7, 1);
   }
void rul_left (void)
   {
      output_bit ( PIN_D2, 1);
      output_bit ( PIN_D3, 0);
      output_bit ( PIN_D4, 0);
      output_bit ( PIN_D5, 0);
      output_bit ( PIN_D6, 0);
      output_bit ( PIN_D7, 1);
   }
void right (void)
      {
```

```
      output_bit ( PIN_D2, 0);
      output_bit ( PIN_D3, 1);
      output_bit ( PIN_D4, 0);
      output_bit ( PIN_D5, 1);
      output_bit ( PIN_D6, 0);
      output_bit ( PIN_D7, 1);
   }
void rul_right (void)
   {
      output_bit ( PIN_D2, 0);
      output_bit ( PIN_D3, 0);
      output_bit ( PIN_D4, 0);
      output_bit ( PIN_D5, 1);
      output_bit ( PIN_D6, 0);
      output_bit ( PIN_D7, 1);
   }
void stopp (void)
      {
      output_bit ( PIN_D2, 0);
      output_bit ( PIN_D3, 0);
      output_bit ( PIN_D4, 0);
      output_bit ( PIN_D5, 0);
      output_bit ( PIN_D6, 0);
      output_bit ( PIN_D7, 1);
   }
void step_r (void)
   {
      OUTPUT_C(0x50);
      delay_ms (15);
      OUTPUT_C(0x60);
      delay_ms (15);
      OUTPUT_C(0x90);
      delay_ms (15);
      OUTPUT_C(0xA0);
      delay_ms (15);
   }
void servo_0 (void)
   {
      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_C3, 1);
         delay_us (650);
         output_bit ( PIN_C3, 0);
```

```
         delay_ms (20);
         servo = 0;
      }
   }
void servo_45 (void)
   {

      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_C3, 1);
         delay_us (1050);
         output_bit ( PIN_C3, 0);
         delay_ms (20);
         servo = 45;
      }
   }
void servo_90 (void)
   {
      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_C3, 1);
         delay_us (1450);
         output_bit ( PIN_C3, 0);
         delay_ms (20);
         servo = 90;
      }
   }
void servo_135 (void)
   {
      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_C3, 1);
         delay_us (1900);
         output_bit ( PIN_C3, 0);
         delay_ms (20);
         servo = 135;
      }

   }
void servo_180 (void)
   {
      for (i=1;i<=23;++i)
      {
```

```
         output_bit ( PIN_C3, 1);
         delay_us (2350);
         output_bit ( PIN_C3, 0);
         delay_ms (20);
         servo = 180;
      }
   }
void servo_0t (void)
   {
      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_A5, 1);
         delay_us (800);
         output_bit ( PIN_A5, 0);
         delay_ms (20);
         servo = 0;
      }
   }
void servo_45t (void)
   {

      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_A5, 1);
         delay_us (860);
         output_bit ( PIN_A5, 0);
         delay_ms (20);
         servo = 45;
      }
   }
void servo_90t (void)
   {
      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_A5, 1);
         delay_us (1070);
         output_bit ( PIN_A5, 0);
         delay_ms (20);
         servo = 90;
      }
   }
void servo_135t (void)
   {
```

```
      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_A5, 1);
         delay_us (1500);
         output_bit ( PIN_A5, 0);
         delay_ms (20);
         servo = 135;
      }

   }
void servo_180t (void)
   {
      for (i=1;i<=23;++i)
      {
         output_bit ( PIN_A5, 1);
         delay_us (2050);
         output_bit ( PIN_A5, 0);
         delay_ms (20);
         servo = 180;
      }
   }
#int_RB
void  RB_isr(void)
{
       if (notprer == 1)
       {
              if (!input(PIN_B4))                          //Tsop dno
              {
                     if (!input(PIN_B5))                   //Tsop krut
                     {
                            if (s0 > 0)
                            {
                                   if (s0 <= 7)
                                   {
                                          output_bit ( PIN_E0, 0);
                                          rul_right();
                                          delay_ms (100);
                                          output_bit ( PIN_E0, 1);
                                          break;
                                   }
                            }
                            if (s0 >= 8)
                            {
```

```
        if (s0 <= 15)
        {
                output_bit ( PIN_E1, 0);
                stop();
                delay_ms (500);
                right();
                delay_ms(300);
                stop();
                delay_ms (200);
                output_bit ( PIN_E1, 1);
                break;
        }
}
if (s0 >= 16)
{
        output_bit ( PIN_E2, 0);
        rul_left();
        delay_ms (100);
        output_bit ( PIN_E2, 1);
        break;
}
if (s180 > 0)
{
        if (s180 <= 7)
        {
                output_bit ( PIN_E0, 0);
                rul_left();
                delay_ms (100);
                output_bit ( PIN_E0, 1);
                break;
        }
}
if (s180 >= 8)
{
        if (s180 <= 15)
        {
                output_bit ( PIN_E1, 0);
                stop();
                delay_ms (500);
                left();
                delay_ms(300);
                stop();
                delay_ms (200);
```

```
                                        output_bit ( PIN_E1, 1);
                                        break;
                                }
                        }
                        if (s180 >= 16)
                        {
                                output_bit ( PIN_E2, 0);
                                rul_right();
                                delay_ms (100);
                                output_bit ( PIN_E2, 1);
                                break;
                        }
                }
        }
        else
        {
                stop();
                back();
                delay_ms (150);
                stop();
                right();
                delay_ms (100);
                stop();
                break;
        }


    }
    break;
}

void main()
{
      notprer = 1;
      program = 0;
        SET_TRIS_B( 0xF0 );
      setup_adc_ports(NO_ANALOGS);
         setup_adc(ADC_CLOCK_DIV_2);
         setup_psp(PSP_DISABLED);
         setup_spi(SPI_SS_DISABLED);
         setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
         setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);
         setup_timer_2(T2_DIV_BY_1,131,1);
```

```
         setup_ccp2(CCP_PWM);
         set_pwm2_duty(264);
         setup_ccp1(CCP_PWM);
         set_pwm1_duty(264);
         setup_comparator(NC_NC_NC_NC);
         setup_vref(FALSE);
         //enable_interrupts(INT_RB);
      //enable_interrupts(GLOBAL);
   /*Ia?aei eiaeeaoee*/
      output_bit ( PIN_E0, 0);     //green
      delay_ms (100);
      output_bit ( PIN_E0, 1);
      output_bit ( PIN_E1, 0);   //blue
      delay_ms (100);
      output_bit ( PIN_E1, 1);
      output_bit ( PIN_E2, 0);   //red
      delay_ms (100);
      output_bit ( PIN_E2, 1);
   /*Eiiao*/
   // TODO: USER CODE!!
while(1)
{

start:  count=0;
               nachalo=0;
               i=0;
if (input(PIN_A4)==1)
{
       while(input(PIN_A4)==1)
       {
               count=0;
       }
}
else
{
       while(input(PIN_A4)==0)
       {
               i++;
               if (i==100)
               {
                       stopp();
                       goto start;
               }
```

```
        }
}
if (i<100)
{
        nachalo=1;
}
nachalo1:               count=0;
                                if(nachalo==1)
                                {
                                        while(input(PIN_A4)==1)
                                        {
                                                count++;
                                                delay_us(200);
                                        }
                                        while(input(PIN_A4)==0)
                                        {
                                                count++;
                                                delay_us(200);
                                                if (count>100)
                                                {
                                                        goto start;
                                                }
                                        }
                                        program=1;
                                }
           r=count;
if (program==1)
{
   if (count==50)
   {
      forward();
   }

   if (count==25)
   {
      back();
   }
   if (count==17)
   {
      right();
   }
   if (count==13)
   {
```

```
      stopp();
   }
}
else
{
   stopp();
}

}
}
```