Адреса редколегії:
Луцький національний технічний університет,
кафедра комп'ютерної інженерії,
вул.Львівська 75, ауд.141
м.Луцьк, 43018
тел. (0332) 74-61-15
E-mail: cit@lntu.edu.ua,
ekaterinamelnik@gmail.com
сайт журналу: **ki.lutsk-ntu.com.ua**

КОМП'ЮТЕРНО-ІНТЕГРОВАНІ ТЕХНОЛОГІЇ: ОСВІТА, НАУКА, ВИРОБНИЦТВО

№20 2015р.

Журнал засновано у грудні 2010 р.
Свідоцтво про реєстрацію КВ № 16705–5277 Р.
Засновник: Луцький національний технічний університет
Рекомендовано до друку Вченою радою
Луцького національного технічного університету
(протокол №13 засідання від 23.06.2015)
Журнал рішенням президії ВАК України
від 30 березня 2011
№1-05/3 включено в перелік наукових фахових видань.
Журнал має російський індекс наукового цитування
(РІНЦ)

ISSN:978-617-672-040-9

ЗМІСТ

Автоматика та управління

Інформатика та обчислювальна техніка

Педагогічні науки

УДК 519.6
Багнюк Н.В., Завіша В.В., Решетило О.М
Луцький національний технічний університет

ГРАДІЄНТНИЙ МЕТОД ЧИСЛОВОЇ ОПТИМІЗАЦІЇ ЗАДАЧ НЕЛІНІЙНОГО ПРОГРАМУВАННЯ

Багнюк Н.В., Завіша В.В., Решетило О.М. Градієнтний метод числової оптимізації задач нелінійного програмування В статті розглянуто застосування градієнтного методу для задач нелінійного програмування з обмеженнями. Наведений приклад.
Ключові слова: оптимізація. градієнтні методи, нелінійне програмування.
Форм. 8. Рис. 1. Літ. 2.

Багнюк Н.В., Завиша В.В., Решетило А.Н. Градиентный метод числовой оптимизации задач нелинейного программирования. В статье рассмотрено применение градиентного метода для задач нелинейного программирования с ограничениями.
Ключевые слова: оптимизация. градиентные методы, нелинейное программирование.

Bagnyuk N.V., Zavisha V.V., Reshetylo A.N. Gradient method numerical optimization for problems of nonlinear programming. The article deals with the application of the gradient method for nonlinear programming problems with restrictions.
Keywords: optimization. gradient methods, nonlinear programming.

Для задач нелінійного програмування не існує універсального методу розв'язання, що зумовило розроблення значної кількості різних методів розв'язання окремих типів задач нелінійного програмування. Відомі точні методи розв'язання нелінійних задач, але в такому разі існують труднощі обчислювального характеру, тому для розв'язання нелінійних задач виправданим є застосування наближених методів.

Градієнтні методи належать до наближених числових методів розв'язування задач нелінійного програмування, оскільки дають точний розв'язок за нескінченне і лише в окремих випадках за скінченне число кроків. З їх використанням можна розв'язувати будь-яку задачу нелінійного програмування, знаходячи, як правило, лише локальний екстремум. Тому застосування цих методів дає найбільший ефект для розв'язування задач випуклого програмування, де локальний екстремум є одночасно і глобальним.

Розглянемо спочатку задачу максимізації функції $f(x)$, коли обмеження на область зміни змінних x відсутні. Пошук екстремального значення функції $f(x)$ можна починати з будь-якого допустимого розв'язку, наприклад, з точки $x_k = (x_{1k}; ...; x_{nk})$.

Градієнтом $\nabla f(x)$ функції $f(x)$ в точці x_k називається вектор, координатами якого є значення в цій точці частинних похідних першого порядку відповідної змінної, тобто

$$\nabla f(x_k) = (\frac{\partial f}{\partial x_{1k}};...;\frac{\partial f}{\partial x_{nk}}).$$

Градієнт функції в цій точці вказує напрямок найшвидшого зростання функції $f(x)$.

Переміщення з точки x_k вздовж градієнту в нову точку x_{k+1} відбувається по прямій, рівняння якої

$$x_{k+1} = x_k + \lambda_k \nabla f(x_k). \quad (1)$$

де λ_k – числовий параметр, від величини якого залежить довжина кроку переміщення $\Delta x_k = \lambda_k \nabla f(x_k)$. Величина λ_k, при якій досягається найбільший приріст функції, може бути визначена з необхідної умови екстремуму функції

$$\frac{d\left[\nabla f(\lambda_k)\right]}{d\lambda_k} = \nabla f(x_{k+1}) \nabla f(x_k) = 0. \quad (2)$$

Чергову точку x_{k+1} визначаємо після обчислення параметру λ_k (для цього підставляємо

значення λ_k в формулу (1) на пошуковій траєкторії). В цій (x_{k+1}) точці знову знаходимо градієнт, а рух відбувається далі по прямій $x_{k+2} = x_{k+1} + \lambda_{k+1}\nabla f(x_{k+1})$ у напрямку нового градієнту $\nabla f(x_{k+2})$ до точки x_{k+2}, в якій досягається найбільше значення функції $f(x)$ в цьому напрямку і т.д. Розв'язування триватиме доти, поки не буде досягнута точка x^*, в якій градієнт функції дорівнює нулю. В цій точці x^* цільова функція $f(x^*)$ і буде набувати максимального значення.

Тепер розглянемо випадок розв'язування задачі нелінійного програмування з обмеженнями. Припустимо, що задача полягає в наступному: необхідно знайти максимум функції $f(x)$ за обмежень

$$a_i x \le b_i, \quad x \ge 0, \tag{3}$$
$$x = (x_1;...x_n), a_i = (a_{i1};...;a_{in})$$

Крім того, будемо вважати, що функція $f(x)$ є *ввігнутою диференційованою функцією.*

При розв'язуванні подібних задач трапляються два випадки:

1) цільова функція має єдиний екстремум, і він знаходиться всередині області допустимих розв'язків. Тоді процес розв'язування задачі (пошук оптимальної точки x^*) нічим не відрізняється від уже розглянутого;

2) цільова функція набуває свого екстремального значення в точці, що знаходиться на границі допустимої області. В цьому випадку послідовність розв'язування задачі наступна. Якщо початкова точка x_k лежить всередині допустимої області (всі обмеження виконуються як строгі нерівності), то переміщуватися потрібно в напрямку градієнту. Але координати чергової точки $x_{k+1} = x_k + \lambda_k \nabla f(x_k)$ повинні задовольняти обмеженням (3), тобто повинні виконуватись нерівності

$$\begin{cases} a_i\left[x_k + \lambda_k \nabla f(x_k)\right] \le b_i \, (i = 1,...,m); \\ x_k + \lambda_k \nabla f(x_k) \ge 0. \end{cases} \tag{4}$$

Розв'язуючи систему (4) лінійних нерівностей, знаходимо проміжок $[\lambda'_k;\lambda''_k]$ допустимих значень параметру λ_k, при яких точка x_1 буде належати допустимій області. Значення λ_k, яке одержується в результаті розв'язування рівняння (2)

$$\nabla f(x_k + \lambda_k \nabla f(x_k) \cdot \nabla f(x_k)) = 0,$$

повинно належати проміжку $[\lambda'_k;\lambda''_k]$. Якщо значення λ_k виходить за межі проміжку, то за λ_k^* приймається λ''_k. При цьому чергова точка пошукової траєкторії опиняється на граничній гіперплощині, що відповідає нерівності системи (4), виходячи з якої при розв'язанні системи отримано значення λ''_k.

Якщо початкова точка x_k лежить на граничній гіперплощині (або чергова точка пошукової траєкторії опинилась на граничній траєкторії), то напрямок переміщення визначається із розв'язку наступної допоміжної задачі математичного програмування:

$$T_k = \nabla f(x_k) \cdot r_k(\max) \tag{5}$$
$$a_i r_k \le 0 \tag{6}$$

для тих i, при яких

$$a_i r_k = b_i, \tag{7}$$
$$|r_k| = 1, \tag{8}$$

де $r_k = (r_{k1};...;r_{kn})$; $|r_k| = \sqrt{\sum_{j=1}^{n} r_{kj}^2}$.

Умова (7) визначає належність точки x_k границі допустимої області. Умова (6) означає те, що переміщення з точки x_k по вектору r_k буде відбуватися всередину допустимої області або по її границі, а умова (8) необхідна для обмеження величини r_k. Для наступної точки пошукової траєкторії

$$x_{k+1} = x_k + \lambda_k^* r_k$$

знаходиться значення параметра λ_k^*. При цьому використовується необхідна умова екстремуму:

$$(\nabla f(x_{k+1}) \cdot r_k) = 0 .$$

Процес розв'язування припиняється при досягненні точки x_{k+1}^*, в якій

$$T_k = \nabla f(x_{k+1}^*) \cdot r_k = 0 .$$

Запропонований метод розглянемо на наступному прикладі.

Приклад 1. Знайти максимум функції $f = 8x_1 + 6x_2 - 2x_1^2 - x_2^2$ за таких обмежень

$$\begin{cases} -x_1 + x_2 \le 1, \\ 3x_1 + 2x_2 \le 6, \\ x_1 \ge 0, \ x_2 \ge 0. \end{cases}$$

Оптимізаційний пошук почати з точки $x_0 = (1,\ 0.5)$.

Розв'язування. Точка x_0 = (1; 0,5) лежить всередині допустимої області, значення функції в точці x_0 $f(x_0)$ = 8,75. За напрямок переміщення в наступну точку $\boldsymbol{x_1}$ приймаємо напрямок градієнту $\nabla f(x) = (8 - 4x_1;6 - 2x_2)$ в точці $\boldsymbol{x_0}$ = (1; 0,5).

Градієнт у точці $\boldsymbol{x_0}$ дорівнює $\nabla f(x_0) = (8 - 4 \cdot 1;6 - 2 \cdot 0.5) = (8 - 4;6 - 1) = (4;5) \ne 0$. Виходячи з цього, можна записати координати наступної точки

$$x_1 = x_0 + \lambda_0 \nabla f(x_0) = (1;0,5) + \lambda(4;5) = (1 + 4\lambda_0;0,5 + 5\lambda_0).$$

Визначимо проміжок допустимих значень для параметру λ_0, при яких точка x_1 буде належати допустимій області. В цьому випадку система нерівностей (4) має вигляд

$$\begin{cases} -1 - 4\lambda_0 + 0.5 + 5\lambda_0 \le 1; \\ 3 + 12\lambda_0 + 1 + 10\lambda_0 \le 6; \\ 1 + 4\lambda_0 \ge 0; \\ 0.5 + 5\lambda_0 \ge 0. \end{cases}$$

З розв'язку цієї системи знаходимо проміжок $[\lambda_0'; \lambda_0''] = [-0,1;0,09]$. Розв'язавши рівняння

$$\frac{d\nabla f(\lambda_0)}{d\lambda_0} = \nabla f(x_1)\nabla f(x_0) = (4 - 16\lambda_0;5 - 10\lambda_0)(4;5) = 16 - 64\lambda_0 + 25 - 50\lambda_0 = 41 - 114\lambda_0 = 0 ,$$

визначимо значення параметру $\lambda_0 = 0,36$, при якому приріст функції Δf досягає найбільшої величини. Але значення $\lambda_0 = 0,36$ не належить проміжку $[\lambda_0'; \lambda_0''] = [-0,1;0,09]$, тому приймаємо $\lambda_0^* = 0,09$.

Нова точка x_1 = (1,36; 0,95) знаходиться на граничній прямій, яка визначається другим обмеженням–нерівністю (тією нерівністю, якій відповідає значення $\lambda_0^* = 0,09$). В точці x_1 значення

функції $f(x_1) = 11{,}98 > f(x_0) = 8{,}75$. Оскільки точка x_1 лежить на граничній прямій, то напрямок переміщення в наступну точку x_2 визначаємо за вектором r_1 (рух в напрямку градієнта виводить з допустимої області). Для визначення координат вектора r_1 запишемо допоміжну задачу (5) – (8):

знайти максимум функції

$$T_1 = \nabla f(x_1) \cdot r_1 = (2{,}56;4{,}1)(r_{11}; r_{12}) = (2{,}56r_{11} + 4{,}1r_{12})$$

за обмежень

$$a_2 r_1 = (3;2)(r_{11}; r_{12}) = 3r_{11} + 2r_{12} = 0;$$

$$|r_1| = \sqrt{r_{11}^2 + r_{12}^2} = 1.$$

Система рівнянь цієї задачі має два розв'язки: (0,5568;–0,8352) і (–0,5568; 0,8352). Підставляючи ці розв'язки у функцію T_1, одержуємо, що максимальне значення функції $T_1 \neq 0$ і досягається при (–0,5568; 0,8352), тобто переміщуватися з x_1 треба вздовж вектору r_1 = (–0,5568; 0,8352) по другій граничній прямій (Рис. 1). Координати наступної точки x_2 дорівнюють

$$x_2 = x_1 + \lambda_1 r_1 = (1{,}36 - 0{,}5568\lambda_1; 0{,}95 + 0{,}8352\lambda_1),$$

а градієнт

$$\nabla f(x_2) = (2{,}56 + 2{,}2272\lambda_1; 4{,}1 - 1{,}67\lambda_1).$$

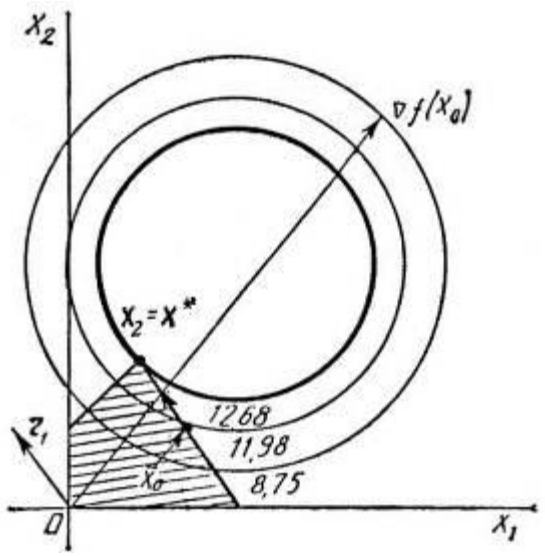


Рис 1. –Графічний розв'язок задачі

Знову визначаємо проміжок допустимих значень параметра λ_1, при яких точка x_2 буде належати допустимій області.

До системи обмежень, які повинна задовольняти точка x_2, не увійде друге обмеження, оскільки ця точка лежить на граничній прямій, визначеній цим обмеженням. Розв'язуючи дану систему, знаходимо проміжок

$$[\lambda_1'; \lambda_1''] = [-1{,}1; 1{,}01].$$

З використанням необхідної умови екстремуму

$$\frac{d\nabla f(\lambda_1)}{d\lambda_1} = \nabla f(x_2) \cdot r_1 = (2{,}56 + 2{,}2272\lambda_1) \times (-0{,}5568) + (4{,}1 - 1{,}6704\lambda_1) \times 0{,}8352 = 0$$

отримуємо $\lambda_1 = 2{,}2$. Але значення $\lambda_1 = 2{,}2$ не належить проміжку $[-1{,}1; 1{,}01]$, тому приймаємо

$\lambda_1^* = 1{,}01$. Нова точка x_2 = (1,36-0,557x1.01; 0,95+0,835x1,01)=(0,8; 1,8) лежить на перетині двох граничних прямих, які відповідають першій і другій нерівностям системи обмежень. У цій точці функція набуває значення $f(x_2) = 12{,}68 > f(x_1) = 11{,}98 > f(x_0) = 8{,}75$.Снову визначимо напрямок переміщення з точки x_2 – вектор $r_2 = (r_{21}; r_{22})$ та розглянемо наступну задачу:

знайти максимум функції

$$T_2 = \nabla f(x_2) \cdot r_2 = (2{,}56 + 2{,}227 \cdot 1{,}01; 4{,}1 - 1{,}67 \cdot 1{,}01)(r_{21} + r_{22}) = (4{,}88 r_{21} + 2{,}4 r_{22})$$

за обмежень

$$r_{21} + r_{22} = 0{,}3 r_{21} + 2 r_{22} = 0;$$

$$|r_2| = \sqrt{r_{21}^2 + r_{22}^2} = 1.$$

Система рівнянь задачі має розв'язок r_2 = (0; 0). Підставляючи одержаний розв'язок у функцію T, дістаємо, що максимум T = 0, а це означає те, що x_2 є точкою максимуму цільової функції в допустимій області, тобто $x_2 = x^*$ і max $f(x^*)$ = 12,68. Як видно з рисунку 1, лінія рівня $f(x)$дотикається до границі допустимої області в точці x_2. Розв'язок знайдено.

Висновки

Сучасні вимоги до математичної підготовки інженера досить високі. Оскільки в більшості практично важливих випадків аналітичне розв'язання задач оптимізації важке або неможливе, необхідно володіти чисельними методами, розрахованими на застосування сучасних комп'ютерів.

В статті розглянуто застосування градієнтного методу, коли наявні обмеження на область зміни змінних x.

1. Нефьодов Ю.М., Балицька Т.Ю. Методи оптимізації в прикладах і задачах. Навчальний посібник. – К.: Кондор, 2011. – 324 с.
2. Ржевський С.В., Александрова В.М. Дослідження операцій: Підручник. – К.: Академвидав, 2006. – 560 с.

УДК 621.317
Білинський Й.Й., Стасюк М.О., Керсов О.П.
Вінницький національний технічний університет

ДОСЛІДЖЕННЯ ВПЛИВУ ТЕМПЕРАТУРИ СЕРЕДОВИЩА НА ТОЧНІСТЬ ДОПЛЕРІВСЬКОГО УЛЬТРАЗВУКОВОГО ВИТРАТОМІРА

Білинський Й.Й., Стасюк М.О., Керсов О.П. Дослідження впливу температури середовища на точність доплерівського ультразвукового витратоміра. Проаналізовано основні недоліки доплерівських витратомірів. Встановлено найбільш суттєві фактори, які впливають на точність вимірювання. Розраховано вплив температури середовища на величину доплерівського зсуву. Запропоновано метод підвищення точності доплерівського ультразвукового витратоміра.
Ключові слова: витратомір, ефект Доплера, доплерівський зсув, швидкість звуку в середовищі, температура середовища.

Билинский Й.Й., Стасюк М.А., Керсов А.П. Исследование влияния температуры среды на точность доплеровского ультразвукового расходомера. Проанализировано основные недостатки доплеровских расходомеров. Определены наиболее существенные факторы, которые влияют на точность измерений. Рассчитано влияние температуры среды на величину доплеровского сдвига. Предложено метод повышения точности доплеровского расходомера.
Ключевые слова: расходомер, эффект Доплера, доплеровский сдвиг, скорость звука в среде, температура среды.

Bilynsky Y.Y., Stasiuk M.O., Kersov O.P. Investigation the influence of temperature of the medium on accuracy of Doppler ultrasonic flowmeter. The basic disadvantages of Doppler flowmeters are analyzed. The most significant factors affecting measurement accuracy are defined. Temperature of the medium effect on the value of the Doppler shift is computed. A method of increasing the accuracy of the Doppler ultrasonic flowmeter is proposed.
Keywords: flowmeter, Doppler effect, Doppler shift, velocity of sound in the medium, temperature of the medium.

Постановка наукової проблеми. Енергозбереження, енергоефективність, оптимальне використання виробничих потужностей і природних ресурсів стали ключовими напрямками розвитку сучасного промислового підприємства. Максимально точне визначення витрати рідких і газоподібних середовищ є одним із головних способів підвищення енергоефективності. Відомі на сьогодні методи вимірювання витрат не є універсальними, кожен з них має свою область застосування.

Основною проблемою комерційних відносин при поставках плинних і газоподібних середовищ є дисбаланс, що виникає при фізичному обліку від постачальника до споживача. Загальними факторами, що визначають виникнення цього дисбалансу, є похибки у вимірюванні об'єму речовини, відсутність достовірного обліку через невисоку точність і обмежений діапазон лічильників, несправності вузлів обліку [1, 2].

Основна проблема практичного використання ультразвукових доплерівських витратомірів пов'язана з тим, що швидкість розповсюдження звуку залежить від таких параметрів середовища, як температура, тиск, концентрація і т. д. [3, 4]. Для рідин швидкість розповсюдження ультразвуку практично залежить лише від температури і концентрації [5]. Проте, найбільший вплив на результати вимірювань має температура, так як навіть невеликі її перепади можуть вплинути на точність. Тому актуальним завданням є дослідження впливу температури на точність доплерівського ультразвукового витратоміра.

Аналіз досліджень.

В [1, 2] описано загальні проблеми сучасних методів контролю витрат.

В [3, 4, 5] наведено основні похибки вимірювання витрат ультразвуковим методом. В [5] проаналізовано похибки ультразвукових витратомірів і запропоновано способи їх усунення. Наведено формули розрахунку сумарних похибок для різних методів ультразвукового контролю витрат.

В [6] наведено основні параметри доплерівських витратомірів. Описана математична модель вимірювання витрат методом Доплера. Наведено швидкості розповсюдження звуку у різних рідина та газах для розрахунку доплерівського зсуву.

В [7] наведено модифіковане рівняння стану реального газу Ван−дер−Ваальса. Виведено залежності констант рівняння від параметрів газу. Практично доведено правильність математичної моделі. Та наведено параметри різних газів, які можуть бути використані для розрахунків.

В [8] експериментально перевірено адекватність розрахунку швидкості розповсюдження звуку у газах. Наведено значення показника адіабати різних газів та рідин при різних температурах та фізичних станах.

В [9] наведено локаційний варіант формули ефекту Доплера.

В [10] проведено порівняльний аналіз ультразвукових витратомірів для чистих і забруднених середовищ. Наведено методику розрахунку акустичного перетворювача та електронних вимірювальних схем ультразвукових витратомірів.

Виклад основного матеріалу й обґрунтування отриманих результатів дослідження.

У більшості випадків п'єзоелементи доплерівських витратомірів розміщуються ззовні труби. Це особливо необхідно у випадку вимірювання забруднених і абразивних середовищ, але необхідно враховувати додаткові похибки, спричинені заломленням хвилі на розділі двух середовищ [5].

Принцип дії доплерівського витратоміра полягає у вимірюванні різниці частот (доплерівського зсуву), яка визначається за формулою:

$$\Delta f = \frac{2 \cdot v_ч \cdot f_1 \cdot \cos\alpha}{c}. \quad (1)$$

де Δf – різниця частоти падаючої і відбитої ультразвукової хвилі, $v_ч$ – швидкість частинки в потоці, f_1 – частота падаючої хвилі, α – кут, під яким опромінюється частинка в потоці, c – швидкість звуку в середовищі.

Ширина доплерівського спектру залежить від діаграми спрямованості, яка формується випромінювачем і приймачем ультразвуку, тобто чим вона ширша, тим ширшим є спектр сигналу.

$$\Delta f = 2 f_1 \sum_{n=1}^{\infty} \frac{2 \cdot v_{чп} \cdot \cos\alpha_n}{c}. \quad (2)$$

На основі теоретичних даних [6] для швидкості розповсюдження ультразвуку у різних середовищах розраховано різницю частот, яка виникає при проходженні через доплерівський витратомір для таких газів, як повітря, хлорин, метан, гідроген та рідин – вода, метанол, керосин, гліцерин. Розраховано чутливості плинних і газоподібних середовищ при зміні швидкості на 1 м/с. На рис.1 подано графіки залежностей, отримані за допомогою розробленої програми на мові Python, різниці частот опромінюючої і відбитої частинкою потоку ультразвукових хвиль від зміни швидкості речовини для різних газів, на рис. 2 − для рідин. Доплерівський витратомір працює при швидкостях потоку речовини від 0,1 м/с до 6 м/с.

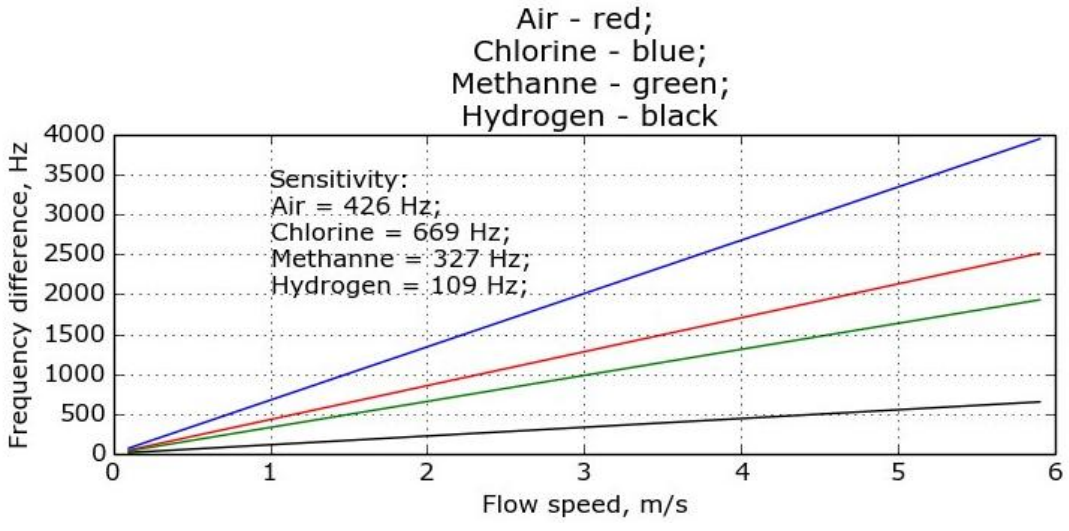


Рис. 1. Залежності різниці частот від зміни швидкості потоку газів від 0,1 м/с до 6 м/с

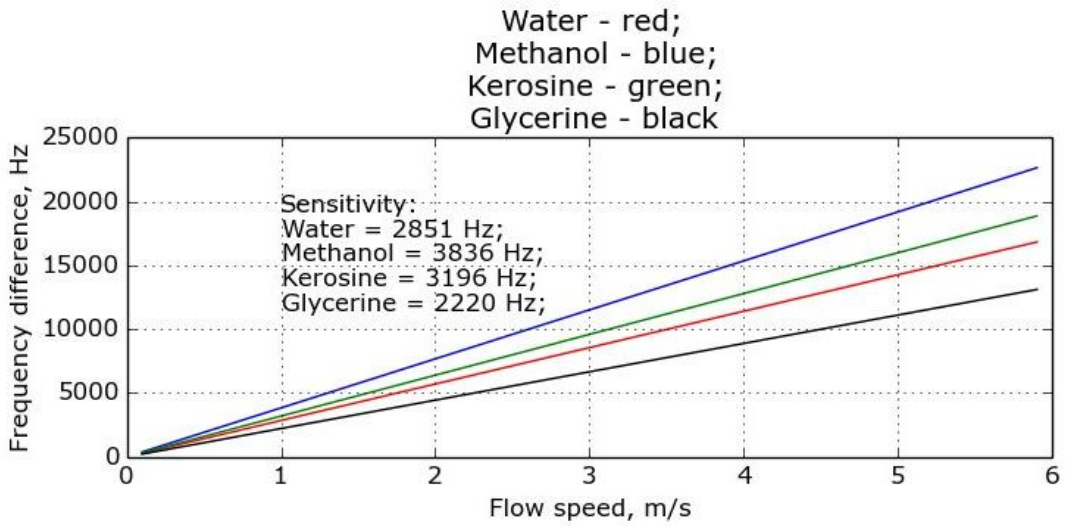


Рис. 2. Залежності різниці частот від зміни швидкості потоку від 0,1 м/с до 6 м/с

Проте, при вимірюванні витрат газів і рідин в реальних умовах існують різні похибки, які залежать від фізико−хімічних властивостей середовища. Сумарна похибка вимірювання витрат для ультразвукових частотних витратомірів розраховується за формулою [5]:

$$Q_O = \frac{\pi D^3 k \mathrm{V} f}{2 \sin 2a}, \tag{3}$$

де $D = L \sin a$, L − довжина акустичного шляху у вимірюваній речовині, $k = v_C / v_D$ − відношення середньої швидкості v_C по січенню до середньої швидкості по діаметру v_D, Δf − різниця частоти падаючої і відбитої ультразвукової хвилі, α − кут, під яким опромінюється частинка в потоці.

З формули сумарної похибки (5) можна зробити висновок, що похибка вимірювання найбільше залежить від довжини акустичного шляху у вимірюваній речовині L та від доплерівської різниці частот Δf, яка у свою чергу залежить від частоти падаючої хвилі f_1 та швидкості розповсюдження звуку у середовищі c. Врахувавши ці параметри у розрахунках можна досягти підвищення точності доплерівського ультразвукового витратоміра.

Обрано конструкцію витратоміра з випромінювальним і приймальним елементами, які розташовані на протилежних поверхнях труби. У цьому випадку довжина акустичного шляху хвилі буде найменшою, а тому відхилення та заломлення хвиль будуть зведені до мінімуму. До того ж паразитні коливання, які розповсюджують по поверхні труби від випромінювача до приймача в обхід плинного середовища будуть затухати через досить велику відстань між п'єзоелементами. Також для усунення паразитних коливань можуть використовуватись різні конструкторські рішення для відведення цих коливань.

Однак, найбільші похибки можуть виникати при вимірюванні доплерівської різниці частот Δf, яка залежить від швидкості розповсюдження звуку у середовищі c. Саме швидкість розповсюдження звуку у середовищі є не табличним значенням і залежить в першу чергу від температури. Швидкість розповсюдження звуку у середовищі розраховується за формулою Лапласа:

$$c = \sqrt{\frac{\gamma R T}{V_m \rho}}, \tag{4}$$

де γ − показник адіабати, R − універсальна газова стала, T − абсолютна температура, V_m − молярний об'єм, ρ − густина.

В свою чергу молярний об'єм V_m розраховується з модифікованого рівняння стану реального газу Ван−дер−Ваальса [7]:

$$RT = \left(V_m - b\right)\left[P + \frac{a}{\left(V_m + c\right)^k T^m}\right], \tag{5}$$

де P − тиск газу, a, b, c − змінні, які залежать від фізико−хімічного стану конкретного газу k, m − табличні дані для різних газів.

На основі аналізу літературних джерел і математичної моделі розраховано доплерівський зсув для повітря і гідрогену при різних значеннях температури і швидкості потоку 3 м/с. Також розраховано зміну доплерівського зсуву при зміні температури на 1 градус. Отримані графіки залежності доплерівського зсуву від температури наведено на рисунках 3, 4.

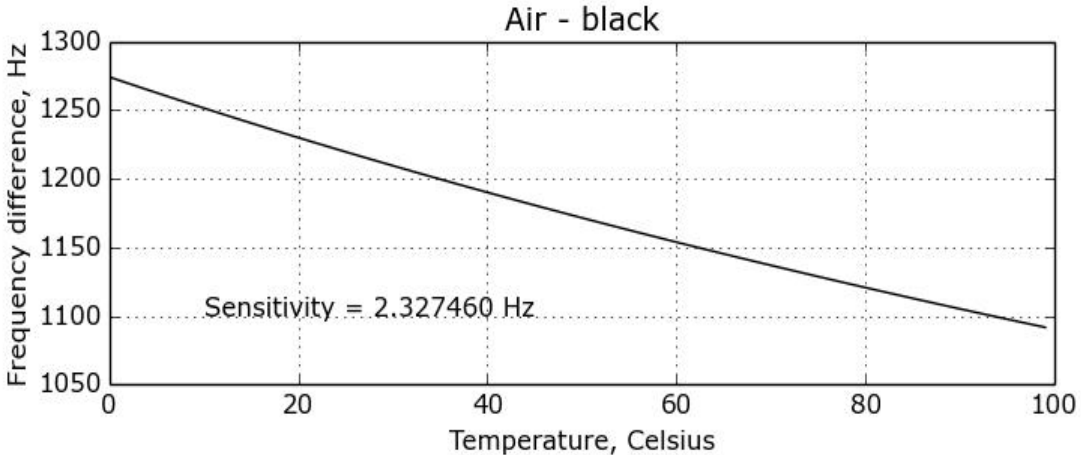


Рис. 3. Залежність доплерівського зсуву від температури повітря

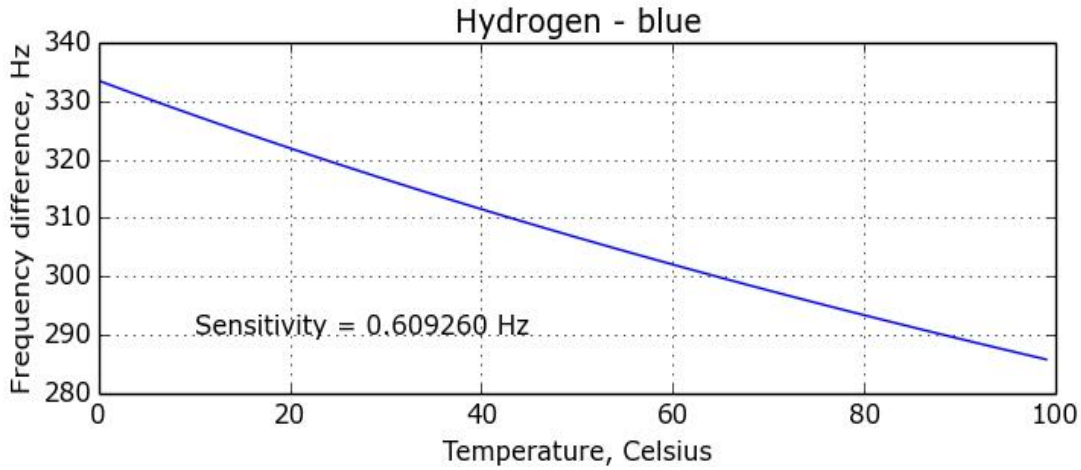


Рис. 4. Залежність доплерівського зсуву від температури гідрогену

Отже, з рис. 3, 4 можна зробити висновок, що зміна температури суттєво впливає на величину доплерівського зсуву. За результатами розрахунків при зміні температури на 1 градус різниця частот змінюється приблизно на 0.2%. Якщо не враховувати температуру газу то похибка при зміні його температури на 10 градусів збільшиться на 1−2 % в залежності від конкретного випадку.

Висновки та перспективи подальшого дослідження. Введення температури газу, як додаткового параметру, в розрахунки витрат для доплерівського ультразвукового витратоміра дозволить суттєво підвищити точність вимірювання при мінімальних затратах на реалізацію. А подальші дослідження впливу фізико−хімічних параметрів плинних середовищ на точність вимірювання витрат дозволять досягти мінімальної похибки.

1. Ряховский С. В. Основные принципы создания единой системы учета газа в региональной компании поставщика газа / С. В. Ряховский, Л. Г. Паскаль. // Энергосбережение. – 2005. – №10.
2. Тюленев Л. Н. Методы и средства измерений, испытаний и контроля / Л. Н. Тюленев, В. В. Шушерин, А. Ю. Кузнецов. – Екатеринбург: ГОУ ВПО УГТУ УПИ, 2005. – 83 с.
3. Хансуваров К. И. Техника измерения давления, расхода, количества и уровня жидкости, газа и пара / К. И. Хансуваров, В. Г. Цейтлин. – Москва: Издательство стандартов, 1990. – 287 с.
4. Друзьякин И. Г. Технические измерения и приборы / И. Г. Друзьякин, А. Н. Лыков. – Пермь: Издательство Пермского государственного технического университета, 2008. – 412 с.
5. Кремлевский П. П. Расходомеры и счетчики количества веществ: Справочник / П. П. Кремлевский. – СПб: Политехника, 2004. – 412 с.
6. Fundamentals of ultrasonic flow measurement for industrial applications [Електронний ресурс] // KROHNE. – 2001. – Режим доступу до ресурсу: http://www.investigacion.frc.utn.edu.ar/sensores/Caudal/HB_ULTRASONIC_e_144.pdf.
7. Фогельсон Р. Л. Уравнение состояния реального газа / Р. Л. Фогельсон, Е. Р. Лихачев. // Журнал технической физики. – 2004. – №7.
8. Frank M. White. Fluid Mechanics / Frank M. White. – New York: McGraw-Hill, 2009. – 885 с.
9. Близнюк В. И. Ультразвуковые расходомеры и система учета на их основе / В. И. Близнюк, В. В. Костылев, В. Л. Сорокопут. // Измерительная техника. – 1998. – №2. – С. 56–57.
10. Биргер Г. И. Ультразвуковые расходомеры / Г. И. Биргер, Н. И. Бражников. – Москва: Металлургия, 1964. – 382 с.

УДК 004.93: 621.313
Болтенков В.А. к.т.н., Аль-Джасри Гамаль Халед Мохаммед.
Одесский национальный политехнический университет

ИССЛЕДОВАНИЕ АКУСТИЧЕСКИХ СИСТЕМ МОНИТОРИНГА ТЕЧЕЙ ТЕПЛОНОСИТЕЛЯ

Болтьонков В.О., Аль-Джасрі Гамаль Халед Мохаммед. Дослідження акустичних систем моніторингу течій теплоносія. Показано переваги акустичних систем моніторингу течі водяного теплоносія в енергоустаткуванні. Дія систем заснована на реєстрації просторово рознесеною системою мікрофонів широкосмугового звукового сигналу, що генерується перегрітим теплоносієм, який витікає через течію. Обробка вимірювальної інформації заснована на TDOA технології. Досліджено два алгоритми оцінювання координат течі рознесеною мікрофонної системою. Порівняння алгоритмів за критеріями точності оцінювання координат і швидкості обчислень проведено шляхом комп'ютерного моделювання.
Ключові слова: моніторинг течій теплоносія, акустичні мікрофонні системи, TDOA технології, псевдообернення матриці.

Болтенков В.А., Аль-Джасри Гамаль Халед Мохаммед. Исследование акустических систем мониторинга течей теплоносителя. Показаны преимущества акустических систем мониторинга течей водяного теплоносителя в энергооборудовании. Действие систем основано на регистрации пространственно разнесенной системой микрофонов широкополосного звукового сигнала, генерируемого перегретым теплоносителем, истекающим через течь. Обработка измерительной информации основана на TDOA технологии. Исследованы два алгоритма оценивания координат течи разнесенной микрофонной системой. Сравнение алгоритмов по критериям точности оценивания координат и скорости вычислений проведено путем компьютерного моделирования.
Ключевые слова: мониторинг течей теплоносителя, акустические микрофонные системы, TDOA технологии, псевдообращение матрицы.

Boltenkov V.A., Al-Jasri Gamal Khaled Mohammed. The study of acoustic coolant leak monitoring systems. The advantages of the acoustic monitoring system leaks water coolant in power equipment has been shown. Their action is based on registration of broadband audio signal generated by superheated coolant flow with spatially separated system of microphones. The measuring information processing is based on TDOA technology. We studied Two algorithms for estimating the leak with microphone system has been studied. Comparison of algorithms for estimating the coordinates by criteria of accuracy and calculations speed has been performed with computer simulation.
Keywords: coolant leak monitoring, acoustic microphone system, TDOA technology, pseudoinverse matrix.

Постановка научной проблемы. В условиях мирового дефицита энергоносителей особое значение приобретают средства их строгой экономии. Одним из эффективных путей такой экономии является применение систем мониторинга и раннего обнаружения течей водяного теплоносителя в энергетических установках. Раннее обнаружение течи и ее локализация позволяют предотвратить дальнейшее развитие аварийной ситуации в теплотехническом агрегате и предотвратить потери теплоносителя, которыми сопровождается течь. Существует ряд методов мониторинга течей в теплоэнергетических установках [1, 2]. На рис. 1 приведена классификация этих методов.

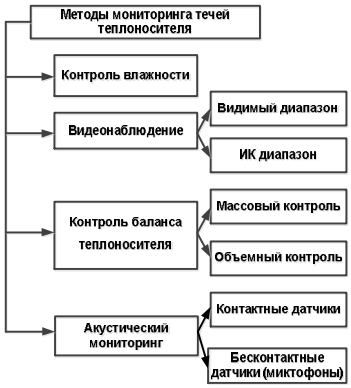


Рис.1. **Классификация методов мониторинга течей теплоносителя.**

Методы контроля влажности основаны на регистрации локального возрастания влажности воздуха в районе течи. Такие методы достаточно чувствительны [3] (их чувствительность по

расходу теплоносителя составляет 5-15 кг/час), но их применение требует расстановки большого числа датчиков влажности в контролируемом помещении теплоэнергетического предприятия и большого числа каналов передачи измерительной информации. Методы видеонаблюдения в видимом или инфракрасном (ИК) диапазоне (тепловидение) наиболее чувствительны: в ИК диапазоне они могут зафиксировать течь с расходом до 1кг/час, однако их практическое применение связано с постоянным визуальным контролем оператора или применением сложных программных систем интеллектуального распознавания образов в регистрируем видеопотоке. Методы контроля баланса теплоносителя – массовые или объемные – (сравнение массы или объема на входе и выходе агрегата) относительно легко реализуются, но их чувствительность крайне низка – 150-200 кг/час. От всех перечисленных недостатков свободны акустические системы мониторинга течей. Акустические системы основаны на регистрации широкополосного акустического сигнала, возникающего при истечении перегретого теплоносителя через дефект в металле. Акустические методы могут быть основаны на размещении на элементах конструкции контактных акустических датчиков, которые регистрируют акустическую волну от течи, распространяющуюся в металле [4].Недостаток таких систем заключается в высокой стоимости контактных акустических датчиков. Наиболее приемлемым вариантом системы мониторинга течей является применение бесконтактных акустических систем, элементами которых являются пространственно разнесенные микрофоны – акустическая сенсорная сеть (АСС) – регистрирующие акустический сигнал от течи, распространяющийся в воздушной среде помещения. АСС достаточно эффективны для обнаружения течей теплоносителя в оборудовании атомных и тепловых электростанций, бойлерных системах и технической диагностики состояния других сосудов, работающих под давлением. Их чувствительность по расходу составляет 20-30 кг/ час. Процесс мониторинга течей легко автоматизируется. Пространственное разнесение микрофонов позволяет не только обнаружить течь, но и оценить местоположение точки истечения теплоносителя, т.е. локализовать течь. Однако сложность обработки измерительной информации и характеристики точности локализации течи в акустических бесконтактных системах изучены недостаточно. Кроме того, важную роль в процессе мониторинга течей играет оперативность контроля, т.е. время, затрачиваемое на оценку координат течи – чем раньше течь будет локализована, тем менее тяжелыми будут ее последствия. Поэтому **целью исследования** является сравнительный анализ алгоритмов локализации течи теплоносителя с помощью акустической микрофонной системы, позволяющих оценить координаты течи с достаточной точностью и с минимальными затратами процессорного времени.

Изложение основного материала и обоснование полученных результатов исследования. Для оценки координат течи предлагается применить TDOA (time difference of arrival) технологии обработки измерительной акустической информации [5]. Они основаны на оценке разности времен прихода сигналов на каждую пару микрофонов. Пусть источник звука расположен в точке S. Тогда микрофона 1, расположенного на расстоянии d_1, сферическая волна достигнет через время $t_1 = d_1 / c$, где c - скорость распространения звука (при нормальных условиях в воздушной среде c =331 м/с), микрофона 2 (d_2) через время $t_2 = d_2 / c$, микрофона 3 (d_3) через время $t_3 = d_3 / c$ (рис. 2).

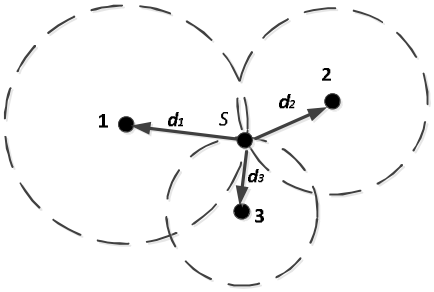


Рис.2. **Принцип действия TDOA системы.**

Поскольку время прихода непрерывного сигнала на микрофоны в пассивной системе измерить невозможно, перейдем к измерению разностей времен прихода. Пусть источник звука расположен в точке с координатами (x_0, y_0, z_0), а микрофоны 1,2,3 – в точках с координатами (x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3) соответственно. Тогда разности времен прихода волны от источника на микрофоны составят:

$$\tau_{12} = 1/c\{ \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2 + (z_0 - z_1)^2} - \sqrt{(x_0 - x_2)^2 + (y_0 - y_2)^2 + (z_0 - z_2)^2} \},$$

$$\tau_{23} = 1/c\{ \sqrt{(x_0 - x_2)^2 + (y_0 - y_2)^2 + (z_0 - z_2)^2} - \sqrt{(x_0 - x_3)^2 + (y_0 - y_3)^2 + (z_0 - z_3)^2} \},$$

$$\tau_{13} = 1/c\{ \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2 + (z_0 - z_1)^2} - \sqrt{(x_0 - x_3)^2 + (y_0 - y_3)^2 + (z_0 - z_3)^2} \}.$$

Получены три уравнения, описывающие двуполостные гиперболоиды вращения. Эти поверхности являются поверхностями положения (источник звука может находиться только на этих поверхностях). Их точка пересечения дает координаты течи. Для двумерного случая требуются три уравнения, для трехмерного - четыре уравнения, т.е. четыре микрофона. Для микрофонной системы, состоящей из N микрофонов можно построить систему уравнений для каждой четверки датчиков:

$$\sqrt{(x_0 - x_i)^2 + (y_0 - y_i)^2 + (z_0 - z_i)^2} - \sqrt{(x_0 - x_j)^2 + (y_0 - y_j)^2 + (z_0 - z_j)^2} = c\tau_{ij},$$

$$\sqrt{(x_0 - x_i)^2 + (y_0 - y_i)^2 + (z_0 - z_i)^2} - \sqrt{(x_0 - x_j)^2 + (y_0 - y_j)^2 + (z_0 - z_j)^2} = c\tau_{ij},$$

$$\sqrt{(x_0 - x_i)^2 + (y_0 - y_i)^2 + (z_0 - z_i)^2} - \sqrt{(x_0 - x_j)^2 + (y_0 - y_j)^2 + (z_0 - z_j)^2} = c\tau_{ij},$$

$$i = 1,...,N,\ j = 1,...,N,\ k = 1,...,N,\ i \neq j \neq k.$$

Решение этой системы дает оценку координат источника звука (течи). В реальных условиях в технологических помещениях присутствуют акустические шумы работающего оборудования, поэтому в оценки разности времен прихода будут внесены случайные погрешности. Разности времен прихода τ_{ij} оцениваются путем определения максимума взаимно-корреляционной функции (ВКФ) сигналов $s_i(t)$ и $s_j(t)$, зарегистрированных парой микрофонов с номерами i, j на интервале наблюдения T :

$$\hat{\tau}_{ij} = \underset{0<t<T}{\arg\max}\ {}^{1}\!/_{T} \int_0^T s_i(t) s_j(t+\tau) dt$$

или в дискретном виде для скользящего окна длиной L

$$\hat{\tau}_{ij} = \underset{0 \le n \le (L-1)}{\arg\max} \sum_{n=0}^{L-1} s_i[n] s_j[n - \Delta\tau].$$

Для получения усредненной статистической оценки координат течи необходимо провести пространственное усреднение по всем N микрофонам системы. Для решения такой задачи возможны две стратегии. Первая состоит в решении системы для каждой четверки микрофонов, не лежащих в одной плоскости, и последующем усреднении. Решение системы гиперболических уравнений, которая является существенно нелинейной, необходимо применение численных методов, например, метода Левенберга-Марквардта. Это достаточно ресурсоемкая вычислительная операция. Например, для системы из 24 микрофонов необходимо решить C_{24}^4= 10626 систем уравнений. Такая стратегия очевидно неприемлема из-за большого времени счета. Вторая стратегия состоит в том, что разности времен прихода рассчитываются относительно одного опорного микрофона , т.е. решается система их $(N-1)$ нелинейных уравнений. Эту стратегию и будем применять в дальнейшем.

Рассмотрим другой TDOA алгоритм оценки координат течи [6]. Пусть $\{(x_m, y_m, z_m)\}_{m=1}^{M}$ - координаты M микрофонов, а (x, y, z) - неизвестные координаты источника. Обозначим через t_m время распространения сигнала от источника до микрофона m , c - скорость распространения

звука, тогда расстояние от источника до микрофона m равно $R_m = ct_m$. Введем $\tau_m = t_m - t_1$- разность времен прихода между микрофонами m и 1 ($\tau_1 = 0$). Тогда

$$c\tau_m = c(t_m - t_1) = R_m - R_1.$$

Отсюда

$$\frac{R_1^2 - R_m^2}{c\tau_m} + 2R_1 + c\tau_m = 0,\ m = 2,3,\text{K},M.$$

Для второго микрофона: $$\frac{R_1^2 - R_2^2}{c\tau_2} + 2R_1 + c\tau_2 = 0.$$

Вычитая это уравнением из аналогичных уравнений для $m = 3,4,\text{K},M$ получим систему уравнений:

$$\frac{R_1^2 - R_m^2}{c\tau_m} - \frac{R_2^2 - R_m^2}{c\tau_m} + c\tau_m - c\tau_2 = 0,\ m = 3,4,\text{K},M. \quad (1)$$

Возведение в квадрат каждого из уравнений дает :

$$R_1^2 - R_m^2 = x_1^2 + y_1^2 + z_1^2 - x_m^2 - y_m^2 - z_1^2 - 2x_1x - 2y_1y - 2z_1z + 2xx_m + 2yy_m + 2zz_m,\ m = 3,4,\text{K},M.$$

В результате подстановки этого результата в (1) имеем:

$$\frac{1}{c\tau_m}(x_1^2 + y_1^2 + z_1^2 - x_m^2 + y_m^2 + z_m^2 - 2x_1x - 2y_1y - 2z_1z + 2x_mx + 2y_my + 2z_mz) -$$

$$-\frac{1}{c\tau_m}(x_1^2 + y_1^2 + z_1^2 - x_2^2 + y_2^2 + z_2^2 - 2x_1x - 2y_1y - 2z_1z + 2x_2x + 2y_2y - 2z_2z) + c\tau_m - c\tau_2 = 0,$$

$m = 3,4,\text{K},M$.

Перепишем последнее уравнение более кратко

$$A_mx + B_my + C_mz + D_m = 0,$$

здесь введены следующие обозначения:

$$A_m = \frac{1}{c\tau_m}(-2x_1 + 2x_m) - \frac{1}{c\tau_2}(2x_2 - 2x_1),$$

$$B_m = \frac{1}{c\tau_m}(-2y_1 + 2y_m) - \frac{1}{c\tau_2}(2y_2 - 2y_1),$$

$$C_m = \frac{1}{c\tau_m}(-2z_1 + 2z_m) - \frac{1}{c\tau_2}(2z_2 - 2z_1),$$

$$D_m = c\tau_m - c\tau_2 + \frac{1}{c\tau_m}(x_1^2 + y_1^2 + z_1^2 - x_m^2 + y_m^2 + z_m^2) - \frac{1}{c\tau_2}(x_1^2 + y_1^2 + z_1^2 - x_2^2 - y_2^2 - z_2^2).$$

Запишем полученную систему из $(M-2)$ уравнений в матричной форме:

$$\begin{bmatrix} A_3 & B_3 & C_3 \\ A_4 & B_4 & C_4 \\ . & . & . \\ . & . & . \\ . & . & . \\ A_M & B_M & C_M \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -D_3 \\ -D_4 \\ . \\ . \\ . \\ -D_M \end{bmatrix}.$$

Для $M \geq 5$ система может быть решена относительно (x, y, z) методом псевдоинверсии матрицы Мура-Пенроуза. Таким образом, путем введения дополнительного опорного микрофона можно решить задачу определения координат источника без трудоемкой вычислительной процедуры

решения системы гиперболических уравнений. Ценой линеаризации задачи является добавление лишнего приемника в элементарную расчетную ячейку.

Для исследования обоих алгоритмов путем компьютерного моделирования разработана моделирующая программа в системе Matlab. Основными функциями программы являются:

– задание числа и координат микрофонов и скорости звука, которая в общем случае является функцией температуры воздуха,

– задание фрагмента записи звукового сигнала, излучаемого источником в виде файла в формате *wav*,

– формирование сигналов, принимаемых каждым приемником, из соображений геометрии задачи и значения скорости звука,

– наложение аддитивного широкополосного гауссова шума на сигнал на каждом приемнике в соответствии с заданным отношением сигнал/шум,

– оценка задержек τ_{ij} по максимуму взаимно-корреляционной функции сигналов $s_i(t)$ и $s_j(t)$, зарегистрированных парой сенсоров с номерами i, j для скользящего окна длиной L

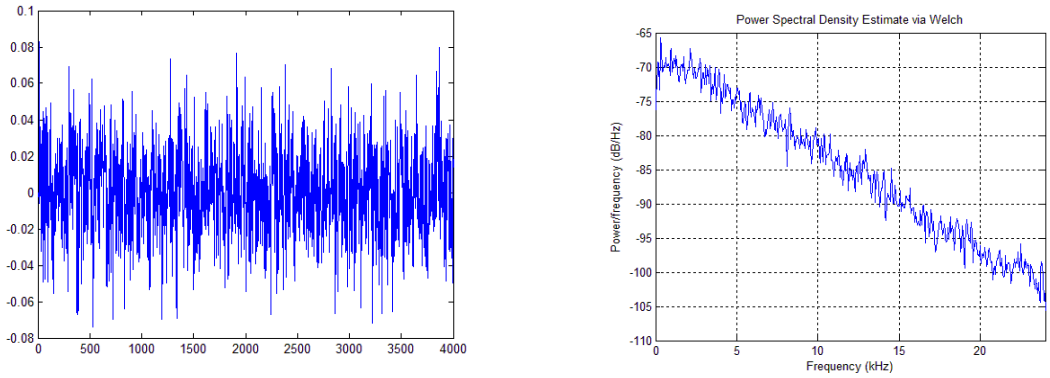
$$\hat{\tau}_{ij} = \arg\max_{0 \le n \le (L-1)} \sum_{n=0}^{L-1} s_i[n] s_j[n - \Delta\tau],$$

– формирование системы гиперболических уранений и ее решение методом Левенберга-Марквардта,

– формирование матричного уравнения и его решение методом псевдоинверсии Мура-Пенроуза,

– расчет среднеквадратичной ошибки оценивания координат источника.

Для моделирования широкополосного акустического сигнала от течи в оборудовании высокого давления использовался розовый шум [7] , т.е. сигнал со спектром вида $1/f^{\alpha}$ (f - частота), $\alpha \approx 1$ в диапазоне (0-20) кГц. Фрагмент сигнала и его спектр показаны на рис.3



а) б)

Рис.3. **Временная картина (а) и спектр (б) акустического сигнала.**

Моделирование проводилось для микрофонной системы, конфигурация которой показана на рис.4. Это АСС из 24-х микрофонов, расположенных 4-мя кольцами по 6 микрофонов, охватывающая зону расположения условного теплотехнического агрегата, в котором может возникнуть течь. Источник звука (течь) S расположен в геометрическом центре системы (координаты источника известны точно). Оценка местоположения течи рассчитывается двумя алгоритмами: путем численного решения системы гиперболических уравнений и путем псевдообращения матрицы с дальнейшим расчетом среднеквадратической ошибки (СКО) оценивания координат источника. Микрофоны включались в систему уравнений последовательно с пяти (начиная с 1_1до 1_5) и до 24-х. Оценивалась СКО оценивания координат и процессорное время счета (при помощи операторов ***tic, toc***). Моделирование осуществлялось на на программно-аппаратной платформе: CPU Intel Core i5 2450M 2.5 ГГц; RAM: 6 ГБ DDR3 1300 МГц; ОС: Win7, Matlab 7.1

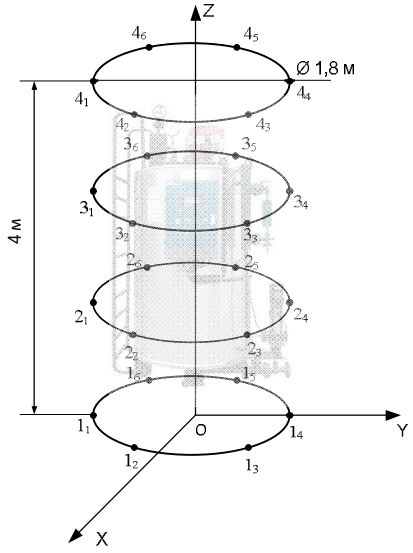
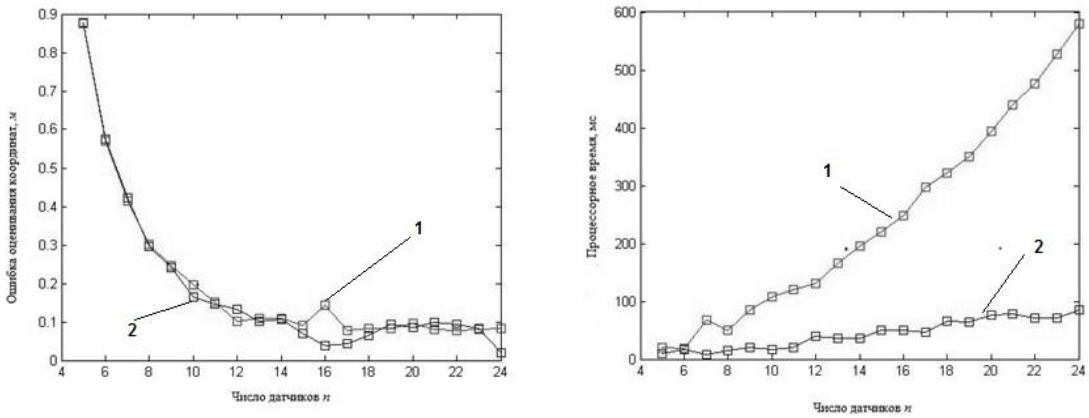


Рис.4. **Конфигурация микрофонной системы.**

Микрофоны вводились в систему уравнений последовательно, начиная с пяти (1_1 ... 1_5) и далее, до 24-х. Оценивалась СКО оценивания координат и процессорное время счета (при помощи операторов ***tic, toc***). Моделирование осуществлялось на на программно-аппаратной платформе: CPU Intel Core i5 2450M 2.5 ГГц; RAM: 6 ГБ DDR3 1300 МГц; ОС: Win7, Matlab 7.1

На рис.5 представлены типичные результаты показателей качества полученные при отношении сигнал/шум, равном 5 дБ (линии 1 относятся к прямому решению системы гиперболических уравнений, линии 2 – к решению путем псевдообращения матрицы.



а) б)

Рис.5. **Показатели качества системы: а) СКО оценивания координат течи, б) процессорное время счета.**

Анализ приведенных результатов моделирования позволяет установить следующее:

1. Точность оценивания координат течи при обоих методах расчета практически одинакова и при отношении сигнал/шум 5 дБ составляет ~ 10 см, что вполне достаточно для практических применений.
2. С ростом числа микрофонов в системе ошибка оценивания координат резко уменьшается при увеличении числа микрофонов примерно до 10, дальнейшее увеличение числа микрофонов в системе несущественно влияет на уменьшение ошибки.
3. Метод псевдоинверсии матрицы позволяет получить существенный выигрыш (почти на порядок) по времени счета по сравнению с решением системы гиперболических уравнений, причем экономия времени расчета растет с увеличением числа микрофонов в системе.

Выводы и перспективы дальнейших исследований. Изложенные результаты исследований позволяют сделать следующие выводы. Акустические микрофонные системы являются весьма перспективными для решения актуальной научной и практической задачи мониторинга течей теплоносителя. Исследованы два алгоритма оценивания координат течи системой разнесенных микрофонов с использование TDOA-технологий: алгоритм, основанный на решении системы гиперболических уравнений, и алгоритм с псевдообращением матрицы. Алгоритмы исследованы с точки зрения точности и оперативности, определяемой временем расчета координат течи. На основе компьютерного моделирования установлено, что для системы из 24 микрофонов СКО оценивания координат течи при отношении сигнал/шум 5 дБ составляет примерно 10 см. При этом алгоритм псевдоинверсии матрицы является существенно более быстродействующим, чем метод, основанный на решении системы гиперболических уравнений.

В качестве направления дальнейших практических исследований следует указать поиск оптимальной конфигурации микрофонной системы и числа ее элементов, обеспечивающих требуемые характеристики качества мониторинга.

1. Geiger, G. State-of-the-Art in Leak Detection and Localization [Text] // Oil and Gas European Magazine. —2006. — No. 4. — P. 193-198.
2. Shahul S. H. Automatic Detection and Analysis of Boiler Tube Leakage System [Text] / S. Shahul Hamid, D. Najumnissa Jamal, Murshitha Shajahan // International Journal of Computer Applications – 2013. – Vol. 84. – No. 16. – P. 19-23.
3. Гетман А.Ф. Концепция безопасности (течь перед разрушением) для сосудов и трубопроводов давления АЭС – М. : Энергоатомиздат, 1999. – 256 с.
4. Маркосян Г.Р. Совершенствование диагностической системы «Alus» для определения места течи теплоносителя из первого контура ВВЭР-440 [Текст] / Маркосян Г.Р. , Петросян В.Г., Шахвердян С.В., Асланян М.А. // Теплоэнергетика .– 2000. - №5. – С.15-20.
5. An L. Hyperbolic boiler tube leak location based on quaternary acoustic array [Text] /Liansuo An, Peng Wang, Augusto Sarti , Fabio Antonacci, Jie Shi// Applied Thermal Engineering. – 2011. – Vol.31 – P. 3428-3436.
6. Steven Li. TDOA Acoustic Localization (2011) [Электронный ресурс].– Режим доступа: http://www.ocf.berkeley.edu/~stevenli/ programming_files / TDOA_Acoustic_Localization.pdf.
7. Болтенков В.А. Алгоритмы обработки информации при акустическом бесконтактном поиске протечек на верхнем блоке реактора ВВЭР-1000 [Текст] / В. А. Болтенков, А. В. Королев, М. В. Максимов, О. В. Маслов. // Известия высших учебных заведений и энергетических объединений СНГ: Энергетика – 2009. – N 3. – С. 67-72.

УДК 539.3
Бортник К.Я., Чухрій С.С
Луцький національний технічний університет

АЛГОРИТМ АВТОМАТИЗОВАНОГО ПРОЇЗДУ В ГРОМАДСЬКОМУ ТРАНСПОРТІ

Бортник К.Я., Чухрій С.С. Алгоритм автоматизованого проїзду в громадському транспорті. Розроблено алгоритм розрахунку пасажирів в громадському транспорті.
Ключові слова: Алгоритм, самообслуговування, електронна оплата квитка, смарт-картка, бортовий автокомп'ютер.

Бортник К.Я., Чухрий С.С. Алгоритм автоматизорованного проезда в общественном транспорте. Разработанный алгоритм расчета пассажиров в общественном транспорте.
Ключевые слова: Алгоритм, самообслуживание, электронная оплата билета, смарт-карта, бортовой автокомпьютер.

Bortnyk K., Chuhriy S. Algorithm for automated public transport. The algorithm of calculating passenger in public transport..
Keywords: Algorithm, self-service, electronic payment card, smart card, board avtokomp'yuter

Для розробки алгоритму передбачаємо, що для комфортного сервісу пасажира в громадському транспорті його необхідно забезпечити сучасними електронними способами оплати проїзду в громадському транспорті (ГТ). Потрібно зауважити, що в даній статті ми не торкаємося таких понять пасажира, як зручність сидінь, температури салону і т. д..

Пасажир повинен мати можливість отримати до проїздки в ГТ інформацію про :

- спосіб проїзду від пункту «А» до пункту «В»;
- графік руху ГТ;
- можливості використання різних видів транспорту для;
- поїздки;
- шлях який потрібно здолати;
- інформацію про пункти проїзду та пересадки ;
- затрачений час на проїзд;
- інформацію про напрямки,повороти шляху, відрізки шляху;
- способи оплати;
- ціну проїзду;
- комфортність сервісу;

і т.д.

Підприємство «Візор-21» вже створило сервіс **«Пошук маршруту»** в комплексі **МАК**, що допомагає пасажиру отримати необхідну інформацію. Розроблені вавігатори ВКС-03 та ВКС-04 з використанням автокомп'ютера (АК) мають змогу забезпечувати наступні сервіси **«Автоматизованого проїзду»** :

- разова оплата проїзду – електронне компостування разового квитка з автоматичним визначенням типу квитка;
- оплата проїзду за допомогою різновиду смарт - карток MIFARE;
- оплата проїзду за допомогою мобільного телефона, смартфона, планшета та інших мобільних пристроїв.

Комплекс **МАК** являє собою цілком український виріб і всі цикли розробки від конструювання, створення зразків на ЗД принтері, лиття форм, розробка та виготовлення електронних плат, написання програм є унікальними.

В даній статті зупинимося на нами розробленому сучасному алгоритмі оплати проїзду «**М**обільна **О**плата **П**роїзду» або «**МОП»**.

Цей алгоритм включає:

- створення системи самообслуговування пасажира;
- здійснення електронної оплати проїзду за допомогою мобільних пристроїв.

МОП може працювати лише при обладнанні ГТ :

- бортовим автокомп'ютером системи МАК;
- Пультом водія;
- клієнт-серверним програмним забезпеченням;
- мобільними каналами зв'язку;
- платіжними сервісами мобільних операторів або програмними додатками системи **МАК**.

Зокрема, Пульт водія має вигляд (рис. 1.)

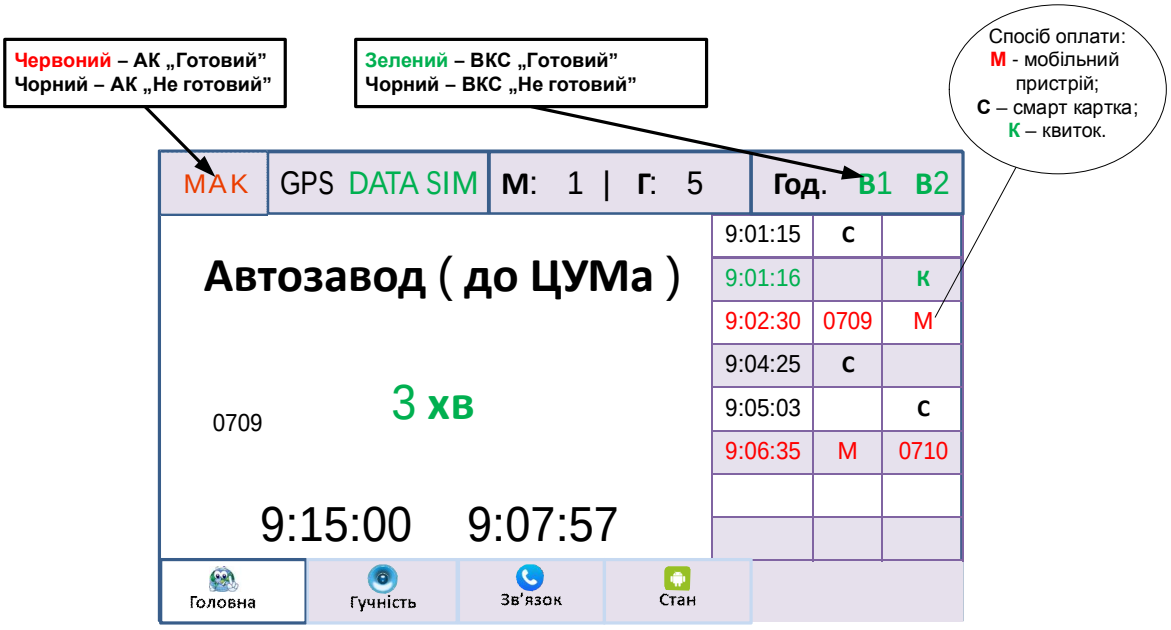


Рис. 1.-- Зображення екрану на Пульті водія.

МОП включає в себе різні мобільні способи оплати, які вибирає пасажир, при допомозі:

1. **Платіжних** систем мобільних операторів;
2. **SMS-оплат** мобільних операторів;
3. Сайту для мобільної оплати проїзду: mobile-mak.lutsk.ua;
4. Створеного **додатку (програми)** для операційних систем мобільних телефонів: Android, Windows Phone, iOS.

Коротко опишемо кожен із способів мобільної оплати:

1. Платіжні системами мобільних операторів:

1.1. МТС – послуга «Мобільні платежі».

http://www.mts.ua/ru/services/upravlenie-schetom/mobilnye-platezhi/mobilnye-platezhi-sovmestno-s-portmone-mobile

Для цієї послуги підходить будь-який мобільний телефон, смартфон, планшет та інші мобільні пристрої.

Для звичайного мобільного телефону здійснення оплати проходить простим набором номера ***135#,*** набором бортового номера ГТ, суми оплати та підтверджуючого коду оператора.

Аналогічно «Мобільні платежі» будуть працювати і для інших мобільних пристроїв.

Послуга безкоштовна.

1.2. Київстар – послуга «Мобільні гроші».

https://money.kyivstar.ua/

Послуга працює аналогічно МТС.

1.3. Лайф – послуга «Мобільні платежі».

https://mpay.life.ua/

Послуга працює аналогічно. МТС

2. SMS оплата мобільних операторів.

Служба коротких повідомлень - за допомогою **SMS** -посилань є можливість здійснювати оплату проїзду. Схоже працює у основних операторів мобільного ринку.

3. Сайт для мобільної оплати проїзду: mobile-mak.lutsk.ua;

Сайт створюється з простим інтерфейсом користувача. В поля вводу вноситься :

- номер телефону з якого необхідно зняти кошти за оплату;
- номер борту;
- тип ТЗ – вибір з випадаючого списку;
- тип білету – вибір з випадаючого списку.

Після вводу вказаних даних приходить **SMS** -повідомлення з кодом підтвердження, який необхідно внести у відповідне поле.

У відповідь приходить СМС з кодом, номером борту, типом білету, який і свідчить про оплату.

Для цієї послуги необхідне підключеня до Інтернету (gprs,wi-fi), смартфон або планшет.

4. Додаток (програма) для Операційних Систем (ОС) мобільних телефонів : Android, Windows Phone, iOS. для Android, Windows Phone, iOS.

Працює лише на смартфонах з вищевказаними ОС. Додаток буде містити аналогічні поля і логіку згідно пункту 3, але оформлені у вигляді окремої програми. Крім того дані програми маю бути завантажені в магазини додатків для кожної ОС відповідно.

Для цієї послуги необхідне підєднання до Інтернет (gprs,wi-fi) та відповідний мобільний пристрій.

Контроль на маршруті. При проведенні контролю на маршруті контролером технологічною карткою контролера блокується робота всіх валідаторів та автокомпютера, пасажир пред'являє прокомпостований в валідаторі квиток або активовану смарт - картку. Контролер візуально оприділяє справжність квитка та перевіряє друковану інформацію, або на електронному пристроїі контролера (ЕПК) перевіряє наявність активації в ТЗ пред'явленої смарт – карти. Для перевірки оплати за допомогою **МОП пасажир має** пред'являти **SMS-** повідомлення з кодом підтвердження оплати проїзду.

Запропонована мобільна система оплати проїзду відповідає сучасному розвитку IT-технологій, забезпечує максимальну зручність пасажирам та є новітньою, яка ще не застосовувалася в ГТ України.

1. http://www.mts.ua/ru/services/upravlenie-schetom/mobilnye-platezhi/mobilnye-platezhi-sovmestno-s-portmone-mobile
2. https://money.kyivstar.ua/
3. https://mpay.life.ua/
4. **mobile-mak.lutsk.ua;**

УДК 514.18
Егорченков В.А.
Национальный авиационный университет, Институт аэропортов, Киев

ПРИНЦИПЫ ПОСТРОЕНИЯ МОДЕЛИ СВЕТОВОЙ СРЕДЫ ПОМЕЩЕНИЙ С КРИВОЛИНЕЙНЫМИ ПОВЕРХНОСТЯМИ

Єгорченков В.О. Принципи побудови моделі світлового середовища приміщень з криволінійними поверхнями. В роботі отримано рівняння, за допомогою якого формується точкова множина для гіперболічної, циліндричної та бочарної поверхонь. Це являється основою для визначення принципів моделювання світлового середовища приміщень з такими покриттями
Ключові слова: сучасна архітектура, точкове числення, світлове середовище, точкове рівняння, криволінійні поверхні, освітленість.

Егорченков В.А. Принципы построения модели световой среды помещений с криволинейными поверхностями. В работе получено уравнение формирования точечного множества для гиперболической, цилиндрической и бочарной поверхностей. Это является основой для определения принципов моделирования световой среды помещений с такими покрытиями
Ключевые слова: современная архитектура, точечное исчисление, световая среда, точечное уравнение, криволинейные поверхности, освещенность.

Постановка проблемы. Начало XXI века характеризуется богатством и разнообразием стилей в городской архитектуре: от классицизма до авангарда. Сегодня встречаются здания самых разных форм: цилиндрических, сферических, многогранных и различных нестандартных объемов.

Необычная, причудливая архитектура зданий, требует нестандартных подходов и конструкторских решений. В них своеобразными являются как световая среда, так и формирующие ее системы естественного освещения.

Одна из основных задач специалистов найти такие принципы моделирования световой среды в таких зданиях, которые позволили бы максимально приблизить параметры среды к оптимальным значениям. Т.е. архитекторам необходимо не только создавать объекты, гармонично вписывающиеся в окружающую среду, но и знать, какие параметры при этом получаются в помещениях и насколько они отличаются от природных.

Анализ последних исследований и публикаций. Классические методы расчета параметров светового поля от какого-либо источника [1] основаны на подсчете величины телесного угла излучаемого элемента методом интегрирования. Этот метод является точным для светопроемов частного положения и простых форм. Но для зданий и проемов не стандартной формы возникают проблемы с определением границ интегрирования. Это обстоятельство снижает точность расчетов и увеличивает компьютерное время работы.

В настоящее время создано ряд мощных программных продуктов для светотехнического проектирования (Lightscape, 3D Studio Viz, Radiance [2] и др.). Однако в расчетных алгоритмах этих программ заложен метод конечных элементов, решение которого осуществляется численно. Если световой элемент находится в плоскости общего положения, то для учёта его положения в пространстве во время расчётов необходимо использование, так называемых, матриц преобразований (матрица поворота, матрица перехода и т.д.), что приводит к неточностям в расчетах, значительно увеличивает время вычислительных операций и нагрузку на центральный процессор и оперативную память.

Формулирование цели статьи. Целью данной работы является разработка принципов построения модели светового поля помещения с криволинейными поверхностями на основе использования БН-исчисления (точечное исчисление Балюбы-Найдыша) [3].

В отличие от других математических аппаратов точечное исчисление позволяет работать непосредственно с объектом, в каком бы пространстве он не находился, а не с его проекциями. Эта его особенность возможна благодаря использованию особых параметров, которые являются инвариантами параллельного проецирования. Такой подход позволяет решить требуемую задачу без использования матриц преобразований.

Основная часть. Представим себе помещение, имеющее покрытие в виде, например, бочарного свода (рис.1).

Определим уравнение гауссовой поверхности через заданные координаты вершин симплекса $ACBB_0$ (пунктирные линии). Зададим опорные контуры поверхности тремя дугами парабол с помощью известных (вид уравнения (3)) точечных уравнений [3]:

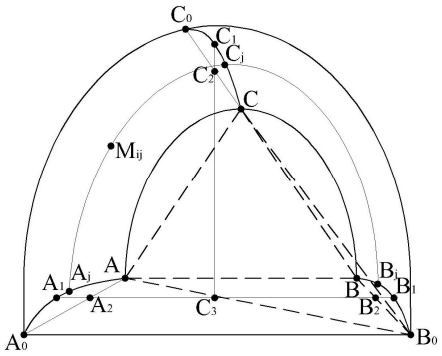


Рис. 1. Расчетная схема помещения с бочарным сводом.

Неизвестные координаты определяются с использованием следующих соотношений, характеризующих показатель кривизны поверхности *w*:

$$\frac{C_1 - B_1}{C - B} = \frac{C_3 - B_1}{C_3 - B_2} = \frac{C_3 - A_1}{C_3 - A_2} = w. \qquad (1)$$

Выразив параметры, входящие в это выражение, через координаты симплекса, получаем следующие уравнения:

$$\begin{aligned} A_j &= A((1-2u)^2 + 2u(1+w)(1-u)) + B_0(1-u) - B(1-u)(2uw+1-2u); \\ B_j &= 2Au(1-u)(1-w) + B_0(1-u) + Bu[2w(1-u) + 2u - 1]; \\ C_j &= 2Au(1-u)(1-w) - B(1-u)(2uw - 2u + 1) + B_0(1-u) + C[(1-2u)^2 + 4uw(1-u)]. \end{aligned} \qquad (2)$$

Здесь A_j, C_j, B_j - текущие точки дуг A_0A_1A, C_0C_1C и B_0B_1B;

$0 \le u \le 1$– параметр, определяющий соответствующие дуги: $u = j/n$, где j = 0, 1, 2, 3,…n.

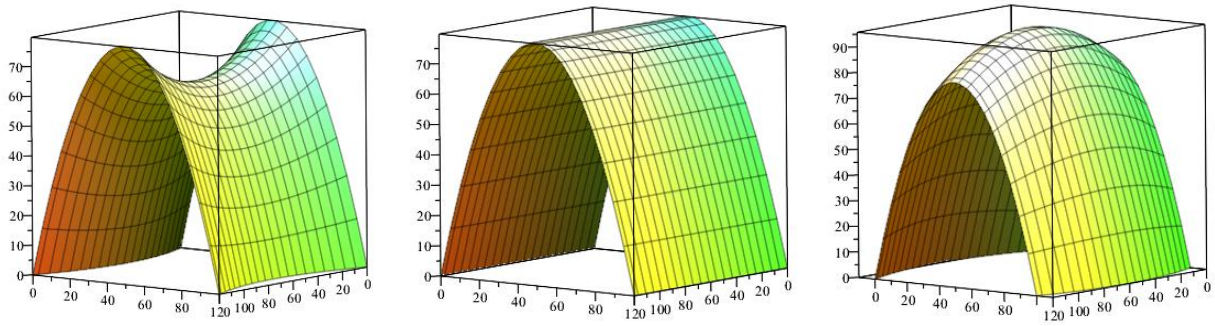
Для определения точечного уравнения поверхности с параболической кривизной составляем уравнение

$$M_{ij} = A_j(1-v)(1-2v) + 4C_jv(1-v) + B_jv(2v-1). \qquad (3)$$

После подстановки выражений (2) в уравнение (3) получаем окончательный вид точечного уравнения исходной поверхности в симплексе $ACBB_0$:

$$\begin{aligned} M_{ij} &= A[(1-v)(1-2v)(1+2u(1-u)(w-1)) + 2uv(1-u)(1-w)(3-2v)] + \\ &+ B[2u(1-u)(w-1)(2v(2v-1)-1) + v(2v-1) - (1-u)] + B_0(1-u) + \\ &+ 4Cv(1-v)[(1-2u)^2 + 4uw(1-u)] \end{aligned} \qquad (4)$$

Реализация этого уравнения в среде Maple представлена на рис. 2.



w=0,8 *w*=1 *w*=1,2

Рис. 2. Визуализация точечного уравнения поверхностей второго порядка:
при w=0,8 – гипар; w=1 – цилиндр; w=1,2 – бочарный свод.

Если четыре соседние точки сканирования принять за основание элементарной пирамиды с известной яркостью, $L_{\varepsilon\upsilon}$, *кд*/*м*^{2}, а ее вершина находится в расчетной точке данной плоскости (рис. 3), то проекция вектора телесного угла на нормаль R к этой плоскости $\sigma_{\varepsilon\upsilon}^{D}$, ср, определится по известной формуле Виннера.

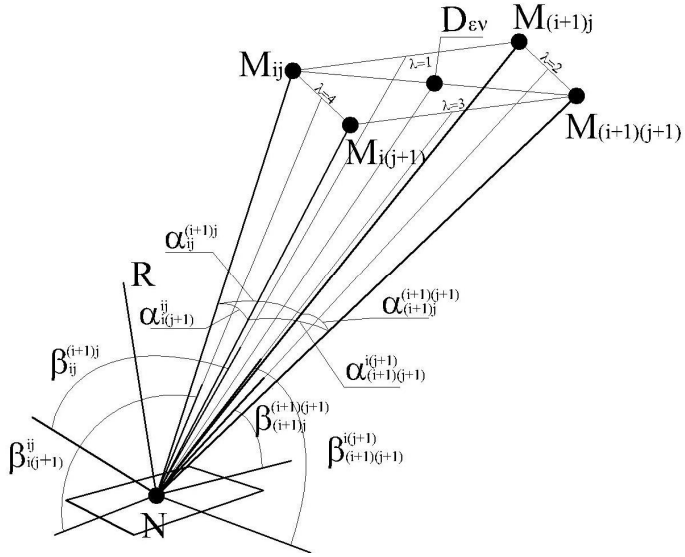


Рис. 3. К определению проекции вектора элементарного телесного угла.

Углы α_{ij} определяются также с использованием точечного исчисления на основе метрических операторов [3] следующим образом (для примера показано определение угловых параметров только для одной грани):

$$\alpha_{ij}^{(i+1)j} = arccos \frac{(x_{ij}-x_N)(x_{(i+1)j}-x_N)+(y_{ij}-y_N)(y_{(i+1)j}-y_N)+(z_{ij}-z_N)(z_{(i+1)j}-z_N)}{\sqrt{\left[(x_{ij}-x_N)^2+(y_{ij}-y_N)^2+(z_{ij}-z_N)^2\right]\left[(x_{(i+1)j}-x_N)^2+(y_{(i+1)j}-y_N)^2+(z_{(i+1)j}-z_N)^2\right]}} \quad (5)$$

Величина $cos\beta_{ij}$ между расчетной плоскостью и гранью элементарной пирамиды определяется как угол между их нормалями. Нормаль к плоскости $M_{ij}M_{(i+1)j}N$ в покоординатном алгоритме определится так

$$\begin{aligned} x_{ij}^{(i+1)j} &= y_{ij}z_{(i+1)j} - y_{(i+1)j}z_{ij} + y_N\left[z_{ij}-z_{(i+1)j}\right] + z_N\left[y_{(i+1)j}-y_{ij}\right];\\ y_{ij}^{(i+1)j} &= z_{ij}x_{(i+1)j} - z_{(i+1)j}x_{ij} + z_N\left[x_{ij}-x_{(i+1)j}\right] + x_N\left[z_{(i+1)j}-z_{ij}\right];\\ z_{ij}^{(i+1)j} &= x_{ij}y_{(i+1)j} - x_{(i+1)j}y_{ij} + x_N\left[y_{ij}-y_{(i+1)j}\right] + y_N\left[x_{(i+1)j}-x_{ij}\right]. \end{aligned} \quad (6)$$

В результате косинус угла между плоскостями определится следующим образом

$$cos\beta_{ij}^{(i+1)j} = \frac{x_R x_{ij}^{(i+1)j} + y_R y_{ij}^{(i+1)j} + z_R z_{ij}^{(i+1)j}}{\sqrt{\left[x_{ij}^{(i+1)j}\right]^2 + \left[y_{ij}^{(i+1)j}\right]^2 + \left[z_{ij}^{(i+1)j}\right]^2}} \quad (7)$$

где x_R, y_R, z_R – координаты точки, определяющей единичную нормаль NR к освещаемой площадке. Они задаются так, чтобы отрезок был перпендикулярен к площадке, а его модуль $\left|NR\right|$=1. Подобным образом определяются косинусы углов для других граней элементарной пирамиды.

Тогда величина проекции вектора элементарного телесного угла находится из следующего выражения

$$\sigma_{\varepsilon\upsilon}^{D} = \frac{1}{2}\left(\alpha_{ij}^{(i+1)j}cos\beta_{ij}^{(i+1)j} + \alpha_{(i+1)j}^{(i+1)(j+1)}cos\beta_{(i+1)j}^{(i+1)(j+1)} + \alpha_{(i+1)(j+1)}^{i(j+1)}cos\beta_{(i+1)(j+1)}^{i(j+1)} + \alpha_{i(j+1)}^{ij}cos\beta_{i(j+1)}^{ij}\right) \quad (8)$$

Освещенность в расчетной точке $E_{\varepsilon\upsilon}^{D}$, *лк*, от участка поверхности в пределах элементарной четырехугольной пирамиды определяется по формуле:

$$E_{\varepsilon\upsilon}^{D} = L_{\varepsilon\upsilon}^{D}\sigma_{\varepsilon\upsilon}^{D} \tag{9}$$

А общая освещенность от всей поверхности в расчетной точке *N* данной плоскости будет равна:

$$E_{N}^{D} = \sum_{\varepsilon=1}^{\nu}\sum_{\upsilon=1}^{\iota} L_{\varepsilon\upsilon}\sigma_{\varepsilon\upsilon}^{D} \tag{10}$$

Величина модуля светового вектора в пределах элементарной четырехгранной пирамиды определится также по формуле (10). Его начало расположено в точке *N*, а направление совпадает с направлением от расчетной точки до центра основания элементарной пирамиды, который может определяться как середина одной из диагоналей

$$\begin{aligned}
xE_{\varepsilon\upsilon}^{D} &= \frac{L_{\varepsilon\upsilon}^{D}\sigma_{\varepsilon\upsilon}^{D}(x_{\varepsilon\upsilon}^{D} - x_{N})}{\sqrt{(x_{\varepsilon\upsilon}^{D} - x_{N})^{2} + (y_{\varepsilon\upsilon}^{D} - y_{N})^{2} + (z_{\varepsilon\upsilon}^{D} - z_{N})^{2}}};\\
yE_{\varepsilon\upsilon}^{D} &= \frac{L_{\varepsilon\upsilon}^{D}\sigma_{\varepsilon\upsilon}^{D}(y_{\varepsilon\upsilon}^{D} - y_{N})}{\sqrt{(x_{\varepsilon\upsilon}^{D} - x_{N})^{2} + (y_{\varepsilon\upsilon}^{D} - y_{N})^{2} + (z_{\varepsilon\upsilon}^{D} - z_{N})^{2}}};\\
zE_{\varepsilon\upsilon}^{D} &= \frac{L_{\varepsilon\upsilon}^{D}\sigma_{\varepsilon\upsilon}^{D}(z_{\varepsilon\upsilon}^{D} - z_{N})}{\sqrt{(x_{\varepsilon\upsilon}^{D} - x_{N})^{2} + (y_{\varepsilon\upsilon}^{D} - y_{N})^{2} + (z_{\varepsilon\upsilon}^{D} - z_{N})^{2}}}.
\end{aligned} \tag{11}$$

А координаты светового вектора от всей поверхности в данной расчетной точке определятся как сумма координат конечных точек векторов всех элементарных пирамид. Имея составляющие светового вектора нетрудно определить величину его модуля. Угловая высота светового вектора (от горизонтальной плоскости) определится

$$\varphi E_{N}^{D} = arctg\frac{ZE_{\varepsilon\upsilon}^{D} - z_{N}}{\sqrt{(x_{\varepsilon\upsilon}^{D} - x_{N})^{2} + (y_{\varepsilon\upsilon}^{D} - y_{N})^{2}}} \tag{12}$$

и азимут светового вектора от оси *Y*

$$\beta E_{N}^{D} = arctg\frac{XE_{\varepsilon\upsilon}^{D} - x_{N}}{YE_{\varepsilon\upsilon}^{D} - y_{N}} \tag{13}$$

Средняя сферическая освещенность в данной точке помещения в пределах элементарного телесного угла определяется по следующей формуле

$$E_{\varepsilon\upsilon}^{4\pi} = 0,25L_{\varepsilon\upsilon}^{D}\omega_{\varepsilon\upsilon}^{D}, \tag{14}$$

где $\omega_{\varepsilon\upsilon}^{D}$ – величина элементарного телесного угла (рис. 4), ср.

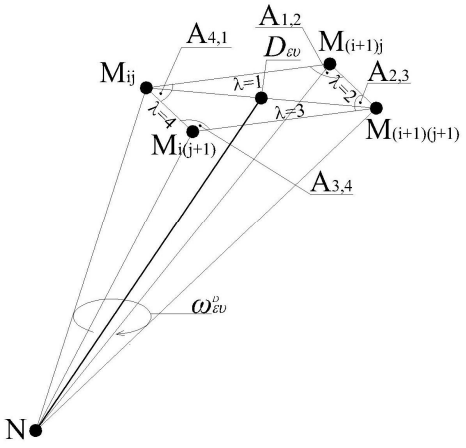


Рис. 4. К определению элементарного телесного угла.

Величина телесного угла, заключенного в пределах граней указанной пирамиды, определяется по следующей формуле [4]

$$\omega_{\varepsilon\upsilon}^{D} = 2\pi - \sum_{k=1}^{p}\left(\pi - A_{k,k+1}\right) \tag{15}$$

где *p* – количество граней элементарной пирамиды;

$A_{k,k+1}$ – величина внутреннего двугранного угла между плоскостями, проходящими через расчетную точку и *k*-тую и *k*+1-ю грани элементарной пирамиды.

Для определения последнего также используется точечное исчисление [3].

Искомый угол между плоскостями определяется как угол между их нормалями. Нормали к плоскостям прово-

дятся из точки *N*. Координаты второй точки нормали к плоскости $M_{ij}NM_{(i+1)j}$, $S_{ij}^{(i+1)j}\left(x_{ij}^{(i+1)j}, y_{ij}^{(i+1)j}, z_{ij}^{(i+1)j}\right)$, определятся из следующих зависимостей:

$$\begin{aligned} x_{ij}^{(i+1)j} &= y_{ij}z_{(i+1)j} - y_{(i+1)j}z_{ij} + y_N\left[z_{ij} - z_{(i+1)j}\right] + z_N\left[y_{(i+1)j} - y_{ij}\right];\\ y_{ij}^{(i+1)j} &= z_{ij}x_{(i+1)j} - z_{(i+1)j}x_{ij} + z_N\left[x_{ij} - x_{(i+1)j}\right] + x_N\left[z_{(i+1)j} - z_{ij}\right];\\ z_{ij}^{(i+1)j} &= x_{ij}y_{(i+1)j} - x_{(i+1)j}y_{ij} + x_N\left[y_{ij} - y_{(i+1)j}\right] + y_N\left[x_{(i+1)j} - x_{ij}\right]. \end{aligned} \quad (16)$$

Координаты нормали ко второй плоскости $M_{(i+1)j}NM_{(i+1)(j+1)}$

$$\begin{aligned} x_{(i+1)j}^{(i+1)(j+1)} &= y_{(i+1)j}z_{(i+1)(j+1)} - y_{(i+1)(j+1)}z_{(i+1)j} + y_N\left[z_{(i+1)j} - z_{(i+1)(j+1)}\right] + z_N\left[y_{(i+1)(j+1)} - y_{(i+1)j}\right];\\ y_{(i+1)j}^{(i+1)(j+1)} &= z_{(i+1)j}x_{(i+1)(j+1)} - z_{(i+1)(j+1)}x_{(i+1)j} + z_N\left[x_{(i+1)j} - x_{(i+1)(j+1)}\right] + x_N\left[z_{(i+1)(j+1)} - z_{(i+1)j}\right];\\ z_{(i+1)j}^{(i+1)(j+1)} &= x_{(i+1)j}y_{(i+1)(j+1)} - x_{(i+1)(j+1)}y_{(i+1)j} + x_N\left[y_{(i+1)j} - y_{(i+1)(j+1)}\right] + y_N\left[x_{(i+1)(j+1)} - x_{(i+1)j}\right]. \end{aligned} \quad (17)$$

В результате угол между плоскостями $M_{ij}NM_{(i+1)j}$ и $M_{(i+1)j}NM_{(i+1)(j+1)}$ определится следующим образом:

$$A_{1,2} = arccos\frac{x_{(i+1)j}^{(i+1)(j+1)}x_{ij}^{(i+1)j} + y_{(i+1)j}^{(i+1)(j+1)}y_{ij}^{(i+1)j} + z_{(i+1)j}^{(i+1)(j+1)}z_{ij}^{(i+1)j}}{\sqrt{\left[x_{(i+1)j}^{(i+1)(j+1)}x_{ij}^{(i+1)j}\right]^2 + \left[y_{(i+1)j}^{(i+1)(j+1)}y_{ij}^{(i+1)j}\right]^2 + \left[z_{(i+1)j}^{(i+1)(j+1)}z_{ij}^{(i+1)j}\right]^2}} \quad (18)$$

Таким же образом вычисляются и другие углы между гранями.

Подставив значения углов в формулу (25) получим величину телесного угла элементарной пирамиды

$$\omega_{\varepsilon\upsilon}^D = A_{1,2} + A_{2,3} + A_{3,4} + A_{4,1} - 2\pi \quad (19)$$

Общее значение средней сферической освещенности получаем суммированием освещенностей от всех пирамид

$$E^{4\pi} = 0,25\left(\sum_{\varepsilon=1}^{\nu}\sum_{\upsilon=1}^{\iota}L_{\varepsilon\upsilon}^D\omega_{\varepsilon\upsilon}^D\right), \quad (20)$$

Таким образом, основные принципы моделирования световой среды помещения, следующие:

1. Определяется точечное уравнения всех поверхностей данного помещения.
2. Формируется точечное множество на этих поверхностях помещения. Определяются элементарные пирамиды.
3. Для каждой пирамиды определяются угловые параметры и проекция вектора телесного угла.
4. Уже известными методами вычисляются освещенности в центре каждой площадки и ее яркость.
5. В пределах элементарной пирамиды определяется элементарная освещенность в расчетной точке, проекции светового вектора, его модуль и угловые параметры.
6. Путем суммирования и итерации определяется общая величина светового вектора и его угловые параметры.
7. Через двугранные углы между гранями элементарных пирамид, вычисляются величины элементарных телесных углов.
8. Определяется средняя сферическая освещенность от каждой площадки.
9. Путем суммирования и итерации определяется общая величина средней сферической освещенности в каждой расчетной точке.

Выводы. В результате определены принципы построения модели светового поля помещений, имеющих покрытие в форме гиперболического параболоида, цилиндра и бочарного свода. Эти принципы могут быть положены в основу разработки программных продуктов для расчета

параметров светового поля таких помещений. Подобные принципы могут быть использованы для формирования моделей светового поля помещений с другими поверхностями.

1. Мешков В.В. Основы светотехники / Мешков В.В. // Учебное пособие для вузов. – М.: Энергия, 1979. – 368 с.
2. Larson G.W. Shakespeare R. Rendering with Radiance: the art and science of lighting visualization. Morgan Kaufmann Publishers. San Francisco. California, 1998. P. 644.
3. Балюба І.Г., Поліщук В.І., Малютіна Т.П. Основи математичного апарату точкового числення / Праці // Таврійська державна агротехнічна академія. Вип. 4. Прикладна геометрія та інженерна графіка. – Т. 29. – Мелітополь: ТДАТА, 2005.– С.22-30.
4. Гершун А.А. Световое поле / Гершун А.А. // Избранные труды по фотометрии и светотехнике. – М.: Физматгиз, 1958.- С. 223 – 397.

УДК 629.113(071):004.01:004.4
Каганюк А.К., Голодюк Н.А.
Луцький національний технічний університет

ВИМОГИ ЩОДО ВІДСТЕЖЕННЯ ТРАНСПОРТНИХ ЗАСОБІВ ЗА ДОПОМОГОЮ РАДІОНАВІГАЦІЙНИХ МЕТОДІВ.

Каганюк О.К., Голодюк Н.А. Вимоги щодо відстеження транспортніх засобів за допомогою радіонавігаційних методів .У статті розглядається опис та засоби радіочастотної ідентифікації рухомого транспорту.
Ключові слова: Радіонавігаційні методи,місцезнаходження,транспортні засоби,радіочастотна ідентифікація

Каганюк А.К., Голодюк Н.А. Требования по обнаружению транспортніх средств с помощью радіонавігационніх методов. В статье рассматривается описание методов радиочастотной идентификации движущего транспорта.
Ключевые слова: Радионавигацыонные методы, местонахождение, транспортные средства, радиочастотная идентификация

Kaganyuk A.K. Golodyuk.N.A. Requirements on finding outтранспортніх facilities by means of радіонавігационніхmethods. Description of methods of radio frequencyauthentication of motive transport is examined in the article.
Keywords: Radionavigayshin methods, location,transport vehicles, radio frequency authentication

Місцезнаходження транспортних засобів, цінних вантажів, рухомих об'єктів набуває все більшої значення в повсіденному житті і є. вкрай **актуальні** при створенні сучасної інфраструктури на сьогодення

Проблема полягає в тому, що необхідно визначитись з класифікацією систем і способів місцезнаходження транспортних засобів.

Підхід, який був рекомендований Міжнародним консультативним комітетом по радіо (МККР) Міжнародного Союзу Електрозв'язку в Звіті 904-1 XVI Пленарної асамблеї (Дубровник, 1986 р.), необхідно переглянути і взявши його за основу, створити сучасне визначення в даному напрямку. Згідно з визначенням, даним в цьому документі, в системах автоматичного (автоматизованого) визначення місцезнаходження транспортного засобу (надалі, слідуючи англомовній абревіатурі, - AVL - Automatic Vehicle Location systems) місце розташування рухомого транспортного засобу, в групі йому подібних, визначається автоматично по мірі переміщення його в межах даної географічної зони.Система AVL звичайно складається з підсистеми визначення місця розташування, підсистеми передачі даних і підсистеми управління та обробки даних. За призначенням AVL системи можна розділити на: **диспетчерські системи,** в яких здійснюється централізований контроль в певній зоні за місцем розташування та переміщенням рухомих об'єктів в реальному масштабі часу одним або декількома диспетчерами системи, що знаходяться на стаціонарних обладнаних диспетчерських центрах, це можуть бути системи оперативного контролю переміщення патрульних автомашин, контролю рухомих об'єктів, системи пошуку викрадених автомашин; **системи дистанційного супроводу,** в яких проводиться дистанційний контроль переміщення рухомого об'єкта з допомогою спеціально обладнаної автомашини або іншого транспортного засобу; найчастіше такі системи використовуються при супроводі цінних вантажів або контролі переміщення транспортних засобів; **системи відновлення маршруту,** вирішальні завдання визначення маршруту або місць перебування транспортного засобу в режимі постобробки на основі отриманих тим чи іншим способом даних; подібні системи застосовуються при контролі переміщення транспортних засобів, а також з метою отримання статистичних даних про маршрутиъ

Конкретні реалізації AVL систем часто включають в свій склад технічні засоби, що забезпечують кілька способів визначення місця розташування. Залежно від розміру географічної зони, на якій діє AVL система, вона може бути: **локальної ,** які. розраховані на малий радіус дії, що характерно в основному для систем дистанційного супроводу;**зональної**, - обмеженої, як правило, межами населеного пункту, області, регіону; **глобальної**, - для якої зона дії складає території декількох держав, материк, територію всієї земної кулі. З точки зору реалізації функцій

місцевизначення AVL системи характеризуються такими технічними параметрами як точність місцевизначення та періодичність уточнення даних. Очевидно, що ці параметри залежать від зони дії AVL системи. Чим менше розмір зони дії, тим вище повинна бути точність місцевизначення. Так, для зональних систем, що діють на території міста, вважається достатньою точність місцевизначення (звана також зоною невизначеності положення) від 100 до 200 м. Деякі спеціальні системи вимагають точності одиниць метрів, для глобальних систем буває досить точності одиниць кілометрів.

Для зональних диспетчерських систем ідеальною може вважатися отримання даних про місцезнаходження рухомого об'єкта до одного разу на хвилину. Системи дистанційного супроводу вимагають більшої частоти оновлення інформації.

Методи визначення місця розташування, використовувані в AVL системах, по класифікації МККР можна розбити на три основні категорії: методи наближення (які у вітчизняній літературі також називаються зоновими методами), методи навігаційного числення і методи визначення місця розташування по радіочастоті.
Нижче розглянуті особливості апаратури і систем місцевизначення, які реально можуть використовуватися в сучасних умовах.

Актуальність наукової роботи полягає в тому, що за останні роки ринок устаткування транспортної і, в першу чергу, автотранспортної електроніки швидко прогресує під впливом розвитку автомобільного ринку – однієї з базових галузей економіки в більшості промислово розвинених країнах. До автомобільної електроніки крім традиційних пристроїв, таких, як процесори управління двигунами і режимами руху сьогодні долучається устаткування електронної навігації, мобільного радіозв'язку та передачі даних. У великих містах Європи, таких, як Париж [1], Брюссель та ін. встановлені сучасні АСУТ, що забезпечує підвищення безпеки пасажирів і дає можливість диспетчерам дотримуватися розкладів руху, своєчасно отримувати і розповсюджувати інформацію про місцезнаходження та стан транспортних засобів, а користувачам міського транспорту мати підвищений комфорт обслуговування.

Супутникова система GPS NAVSTAR все більше використовується як геодезичними, так і іншими службами для розв'язання різноманітних задач геодезії, навігації та транспорту. Для забезпечення потреб споживачів у наш час у світі використовують неперервно діючі референцні (permanent reference GPS stations) або перманентні супутникові радіонавігаційні станції (ПСРНС). Розроблення науково обґрунтованої програми згущення існуючої мережі ПСРНС в Україні дасть можливість максимально ефективно як з геодезичної та навігаційної, так і з економічної точки зору встановлювати нові станції. Під час визначення координат пунктів відносно ПСРНС диференційними GPS - вимірюваннями існує проблема попереднього розрахунку точності визначення компонент векторів. Встановлення зв'язку між параметрами геометричної конфігурації сузір'я супутників GDOP (Geometric Delution Of Precision), тривалістю спостережень, довжиною векторів та точністю GPS - вимірювань дало б змогу прогнозувати точність визначення векторів та використовувати їх для оптимізації побудови GPS - мереж. Розроблення науково – обґрунтованої технології оптимального проектування перманентних супутникових радіонавігаційних мереж (ПСРНМ) дозволить досягти економії витрат на розвиток геодезичних мереж при використанні їх геодезичними та іншими службами.

Проблемою залишилось питання вибору оптимальних робочих частот для конкретного сеансу радіозв'язку, тому що розповсюдження хвиль в даному діапазоні залежать від стану іоносфери, що суттєво впливає на якість і надійність радіозв'язку. В останні роки з'явились модеми декаметрового діапазону, що здатні передавати інформацію по радіоканалу зі швидкістю 9,6 кбіт/с та відносно невеликою (до 9 10–) ймовірністю помилки та відродили зацікавленість до короткохвильового радіозв'язку. Такі світові лідери як Harris, Codan, Motorolla та деякі вітчизняні підприємства почали виробництво радіостанцій, що реалізують алгоритми (ALE – Automatic Link Establishment), що базуються на повторному запиті фрагментів повідомлень та виправленні помилок у масштабі реального часу з мінімальним втручанням людини в цей процес. **При цьому, при розробці нашої системи,** висувається вимога максимальної простоти і надійності реалізованих технічних засобів. Явним фактором, перешкоджаючим тривалому використанню СРНС являється обмеженість їх функціональних можливостей.

На вирішення даної задачі, були направленні зусилля більшого числа вчених і наукових колективів . Основною метою їхньої роботи є розроблення теоретичних основ і практичних рекомендацій оптимального проектування мережі ПСРНС на території України для забезпечення потреб геодезії, аерофотогеодезії, кадастру, транспорту, муніципальних та рятувальних служб, а також прогнозування точності та ефективне застосування вимірювань у режимі реального часу відносно ПСРНС. Для досягнення мети в роботі розв'язуються такі задачі:

1. Розроблення принципів, методики та алгоритму оптимального проектування GPS - мережі ПСРНС України.
2. Теоретичне обгрунтування та розроблення методики визначення інтегрального критерію конфігурації сузір'я супутниківIGDOP.
3. Встановлення функціональної залежності точності GPS - вимірювань від значень IGDOP, довжин векторів та тривалості спостережень.
4. Доведення економічної ефективності побудови ПСРНМ станцій в Україні для її основних споживачів.

До основних критерій РНС відносять :

— зону (область) чи дії робочу зону системи, задану сектором огляду (пошуку) по вимірюваних параметрах об'єкта;

— час огляду (пошуку) заданого чи сектора швидкість огляду;

— обумовлені параметри (координати), їхнє число і точність виміру;

— здатність системи розрізняти об'єкти ;

— пропускну здатність;

— завадостійкість;

— надійність.

Оскільки ці параметри широко використовують для оцінки якості функціонування різних систем, варто дати їхні загальні визначення, що надалі можуть бути уточнені стосовно до конкретних типів РНС.

Зоною дії називають область простору, у якій система надійно виконує функції, що відповідають її призначенню.

Границі робочої зони РНС характеризуються припустимими погрішностями місцезнаходження об'єкта при заданому рівні перешкод.

Майже завжди одним з параметрів, що визначають робочу зону, є дальність дії системи.

Під дальністю дії системи розуміють максимальна відстань, на якому забезпечується одержання заданих показників системи. Найчастіше максимальна дальність дії системи залежить від припустимої похибки при вимірі координат і параметрів руху об'єктів. Під дальністю дії виявлення мають на увазі максимальну дальність, на якій відношення сигналу до шуму ще досить для його виявлення з заданою імовірністю. Іноді зона дії системи обмежена з боку мінімальних значень. У цьому випадку система характеризується двома параметрами: мінімальною і максимальною дальністю дії.

Часом огляду (пошуку) називають час, необхідне для однократного огляду заданої зони дії системи. Вибір часу огляду зв'язаний з маневреністю що спостерігаються чи керованих об'єктів, обсягом простору огляду, рівнем сигналу і перешкод, а також поруч тактичних і технічних характеристик системи.

Число вимірюваних координат, так само як і точність їхнього виміру, визначає можливості системи при її практичному використанні.

Точність системи характеризується погрішностями при вимірі координат і параметрів руху об'єкта. Причинами погрішностей є недосконаліть застосовуваного методу виміру й апаратури, вплив зовнішніх умов і радіоперешкод, суб'єктивні якості оператора, якщо процеси одержання і реалізації інформації не автоматизовані. Вимоги до точності системи залежать від її призначення. Невиправдане завищення вимог до точності приводить до ускладнення системи, зниженню її економічності, а іноді і надійності функціонування.

Здатністю системи розрізняти об'єкти називають здатність роздільного виміру параметрів двох чи декількох близько розташованих у просторі чи об'єктів роздільного керування ними. Відповідно розрізняють здатність, що дозволяє, по дальності і кутових координатах, а також по

відповідним складовим швидкості. Здатність, що дозволяє, кількісно прийнято оцінювати мінімальною різницею значень вимірюваних параметрів сусідніх об'єктів, при якій вони сприймаються системою роздільно.

У радіонавігації звичайно знаходять власні координати об'єкта (єдиного для вимірника) і поняття розрізняти об'єкти часто зв'язують з можливістю визначення сигналу, що несе корисну інформацію про місце об'єкта, з різними паразитними сигналами (відображеннями від іоносфери, місцевих предметів і т.п.), подібними за формою корисному, але достовірної інформації про обумовлені координати не утримуючими.

Пропускна здатність характеризується числом об'єктів, що обслуговуються системою чи одночасно в одиницю часу. Пропускна здатність залежить від принципу дії системи і ряду її тактичних і технічних параметрів і, зокрема, робочої зони, точності і здатності, що дозволяє. Так, РНС, у яких використовується одна лінія зв'язку (різницево-дальномірні чи кутомірні радіомаячного типу), володіють необмеженою пропускною здатністю, тому що можуть одночасно обслуговувати будь-яке число об'єктів.

Пропускна здатність дальномірних систем, заснованих на принципі запиту й активної відповіді (дві лінії зв'язку), обмежена відповідачем, у якому для формування відповідного сигналу на кожен запит необхідно якийсь час. У цьому випадку пропускну здатність характеризують імовірністю обслуговування заданого числа об'єктів при заданому періоді повторення запитів кожним з об'єктів, що знаходяться в робочій зоні системи.

Завадостійкістю РНС — здатність надійного виконання заданих функцій в умовах впливу ненавмисних і організованих перешкод. Перешкодозахищеність визначається скритністю роботи системи і її завадостійкістю.

Під скритністю системи розуміють показник, що характеризує труднощі виявлення її роботи і виміри основних параметрів випромінюваного радіосигналу, а отже, і створення спеціально організованих (прицільних) перешкод. Скритність забезпечується застосуванням гостронаправленого випромінювання, використанням шумоподібних сигналів з низьким рівнем потужності, зміною основних параметрів сигналу в часі.

КількіснюоцінкоюзавадостійкостіРНС є відношення сигналу до перешкоди на вході приймача, при якому похибка виміру заданого параметра не перевершує припустимої з необхідною імовірністю; при цьому повинне забезпечуватися виявлення сигналу з заданою імовірністюпри припустимих значеннях імовірності помилкової тривоги. Необхідна завадостійкість досягається раціональним вибором параметрів радіосигналу системи, а також характеристик ДНА і пристроїв прийому й обробки сигналу.

Основну увагу необхідно приділити надійності та працездатності системи. Надійність — властивість об'єкта зберігати в часі у встановлених межах значення параметрів, що характеризують здатність виконання необхідних функцій у заданих режимах і умовах застосування, збереження і транспортування. Це визначення надійності за ДСТ 27002— 82 є універсальним і цілком відноситься до РНС і пристроїв, з яких вони складаються.

У залежності від причин, що викликають відмовлення в роботі системи, розрізняють наступні різновиди надійності:

— апаратурну, зв'язану зі станом апаратури;

— програмним, обумовленим станом програм обчислювальних пристроїв, використовуваних у системі;

— функціональну, тобто надійність виконання окремих функцій, покладених на систему, і, зокрема, витяги й обробки інформації. У цьому змісті перешкодозахищеність також може бути віднесена до функціональної надійності радіосистеми.

Економічні показники системи, маса і габарити складових її пристроїв є важливими параметрами, що впливають на сукупну оцінку якості системи.

До основних технічних характеристик радіосистеми відносяться параметри, що безпосередньо визначають її тактичні характеристики. Стосовно РНС основними технічними характеристиками є:

— метод огляду (пошуку) і виміру координат і параметрів руху об'єкта;

— робочі частоти, стабільність, потужність, вид модуляції, ширина спектра випромінюваних коливань;

—форма, ширина, коефіцієнт спрямованості антени;

—чутливість і смуга пропущення прийомного пристрою;

—вид і параметри пристроїв відображення і знімання інформації;

—габарити і маса пристроїв, що складають систему, споживана ними енергія від джерел харчування.

Надалі взаємозв'язок тактичних і технічних характеристик буде розглянута для конкретних типів РНС.

Новим в рішенні проблеми є інтелектуальні транспортні системи, які з'явилися не так давно, але розвиток їх концепцій можна прослідкувати починаючи з 70-х років минулого століття, на які припадає період розвитку перших ІТС в Японії.

Вона зорієнтована на інформаційне забезпечення усіх суб'єктів сучасних транспортних комунікацій: власники вантажу (вантажовідправники), автотранспортні підприємства, водії, менеджери страхових компаній, екологічні та санітарні інспекції тощо. Базовою компонентою більшості систем диспетчеризації транспортом є система "автоматизованого місцезнаходження транспортного засобу – АМТЗ" (*Automatic Vehicle Location – AVL*). Система АМТЗ надає можливість диспетчерському центру у реальному масштабі часу слідкувати за місцезнаходженням та графіком руху транспортних засобів, оперативно контролювати виконання завдання та при необхідності перерозподіляти їх на різних маршрутах і напрямах, надавати при необхідності технічну, медичну або іншу допомогу.

Інтерес до СРНС викликаний їх універсальністю. У рамках однієї системи можливе рішення великого комплексу різних задач.

Найбільш перспективними є СРНС "NAVSTAR" (США) і "ГЛОНАСС" (Росія).

На сучасному етапі інтенсивно обговорюються перспективи створення інших супутникових систем типу GPS: Глобальна європейська геостаціонарна система (EGNOSS) та GALILEO. Асоціація європейських авіакомпаній (AEA) виправдали користь останньої системи, рахуючи EGNOSS занадто дорогою у порівнянні з її характеристиками.

В даний час вважається доцільним введення до складу СРНС регіональних додаткових систем, що забезпечують реалізацію найбільш суворих вимог споживачів. Ці структури дозволяють істотно підвищити точність обсервацій, виявляти і ідентифікувати порушення в режимах роботи СРНС, неприпустиме погіршення якості її функціонування та своєчасно попереджати про це споживачів, тобто вони можуть здійснювати контроль цілісності системи та підтримувати режим диференційних вимірювань.

На основі аналізу тенденцій розвитку навіга- ційної апаратури (НА) користувачів радіонавігацій- них систем (1 – 6) можливе визначення наступних напрямів розвитку (НА):

1. Удосконалення характеристик апаратури:
   - 1.1 підвищення характеристик точності;
   - 1.2 підвищення надійності, завадостійкості та електромагнітної сумісності;
   - 1.3 забезпечення автономних методів контролю цілісності системи;
   - 1.4 поширення переліку сервісних завдань;
   - 1.5 зменшення габаритних характеристик;
   - 1.6 зменшення вартісні апаратури.
2. Поширення функціональних можливостей апаратури:
   - 2.1 забезпечення можливості взаємодії апара- тури з автоматизованими інформаційними система- ми та системами управління рухом;
   - 2.2 забезпечення можливості комплексування апаратури з автономними навігаційними системами об'єкта;
   - 2.3 знаходження кутів орієнтації в просторі, поправок системи курсу та інші.
3. Спеціалізація апаратури за наступними ти- пами:
   - 3.1 військова;
   - 3.2 загального призначення;
   - 3.3 спеціальна.

4. Створення уніфікованих функціональних елементів, вузлів, блоків.

ВисновкиНавіть короткий огляд методів і апаратури місцевизначення дозволяє зробити висновок, що не існує універсальної системи, здатної задовольнити всі вимоги кінцевого користувача. Завдання створення ефективно працюючих систем місцевизначення виявляється набагато ширше вибору конкретного методу. Можна виділити наступні проблеми загальносистемного плану, які необхідно враховувати замовникам і розробникам подібних систем. Велике значення має наявність на передбачуваної території розгортання системи відповідної інфраструктури для створення підсистеми передачі даних. Так, наявність системи обчислення і широкомовної передачі корегуючої інформації для роботи навігаційної апаратури в диференціальній режимі (аналогічній, наприклад, радіомаяковой системі Служби берегової охорони США) дозволить значно підвищити точність місцевизначення з використанням СРНС без значного ускладнення бортового обладнання. Наявність систем мобільного зв'язку з стільникового і мікростільниковою структурою дозволить зменшити потужність бортового передавача, що скорочує габарити устаткування, спрощує питання енергозабезпечення (особливо в режимах прихованої установки), ускладнює виявлення бортового устаткування зловмисниками. У свою чергу мікростільникова структура систем зв'язку може стати основою для побудови зонових систем місцевизначення або дозволить вирішувати питання місцевизначення 'радіопеленгаціоннимі "методами. Окремо стоять питання створення електронних карт, призначених для експлуатації з AVL системами, їх актуалізації. Найчастіше геоінформаційні системи, що застосовуються для вирішення завдань місцевизначення, окрім звичайних функцій відображення повинні виконувати функції коректування даних, перерахунку даних, отриманих в різних системах координат, логічної прив'язки траєкторій руху мобільних об'єктів до елементів транспортної мережі з урахуванням моделі руху мобільного об'єкту. З цієї точки зору переваги матимуть ті системи, в яких організована оперативна корекція дорожньої обстановки, аж до обліку інформації про пробки на окремих ділянках транспортних магістралей.

Висновок

Навіть короткий огляд методів і апаратури місцевизначення дозволяє зробити висновок, що не існує універсальної системи, здатної задовольнити всі вимоги кінцевого користувача. Завдання створення ефективно працюючих систем місцевизначення виявляється набагато ширше вибору конкретного методу. Можна виділити наступні проблеми загальносистемного плану, які необхідно враховувати замовникам і розробникам подібних систем. Велике значення має наявність на передбачуваної території розгортання системи відповідної інфраструктури для створення підсистеми передачі даних. Так, наявність системи обчислення і широкомовної передачі корегуючої інформації для роботи навігаційної апаратури в диференціальній режимі (аналогічній, наприклад, радіомаяковой системі

1. Волчко П.І., Іванов В.І., Корольов В.М. та інші "Вимоги до характеристик навігаційної інформації і систем навігації наземних рухомих об'єктів у сучасному штатному процесі", - Сучасні досягнення геодезичної науки та виробництва, №5, стор. 280-283, Ліга-Прес, Львів, 2000.
2. "Global radionavigation – the next 50 years and beyond". Benkers John M. J., Navigation. 2000. 53, №2, стор. 207-214, 1іл, Бібл. 6.
3. "La navigation par satellite, le point de vue des utilisateurs europeens". Bara J. M., Navigation (France). 2000. 48, №191, стор. 69-75.
4. Волчко П.І., Корольов В.М., Макаревич В.Д. та інші "Місце геоінформаційних технологій на базі навігаційної інформації в системах управління взаємодією у підрозділах сухопутних військ", - III Міжнародна науково-технічна конференці "Гіротехнологія, навігація, керування рухом і конструювання рухомих об'єктів", стор. 187-192, Київ, 2001.
5. Карпінський Ю. О., Лященко А. А., Кібець О. Г., Рябчій В. В. Функції та геоінформаційне забезпечення інтелектуальних транспортних систем. //Вісник геодезії і картографії. – 2004. – № 3.– С. 71–79.
6. Harley J. Miller, Shih Lung Shaw. Geographic information systems for transportation: principles and applications. – USA, NY, Oxford University Press, Inc. – 2001. – 460 p.
7. ISO/TR 14825. Geographic Data Files (GDF) – ISO/TC 204/WG3. – 1996. – P. 11–15.
8. ISO/Draft International Standard: GDF – Geographic Data Files. – Version 4.0 – ISO/TC 204/WG3: CD. – 2001. – P. 02–14.
9. Карпінський Ю. О., Дроздівський О. П. Основні принципи побудови базової моделі дорожньої мережі в міжнародному стандарті GDF 4.0. // Зб. наук. праць. Сучасні досягнення геодезичної науки та виробництва. – Львів: НУ "Львівська політехніка", 2005. – С. 302–306.

10. Карпінський Ю. О., Лященко А. А. Формування національної інфраструктури просторових даних – пріоритетний напрям топографо-геодезичної та картографічної діяльності. // Вісник геодезії і картографії. – 2001. – № 3. – С. 65–73.
11. Романишин І. Методика проектування мережі перманентних станцій в Україні // Збірник наукових праць міжнародної науково-практичної конференції "Новітні досягнення геодезії, геоінформатики та землевпорядкування – Європейський досвід", Чернігів, -2005, -С.18-21.
12. Романишин І.Б. До питання попереднього розрахунку точності диференційних GPS – вимірів // Науково-технічний журнал "Сучасні досягнення геодезичної науки і виробництва", -Львів, Ліга-Прес, -2005(2), -С. 120-124.
13. Третяк К.Р., Романишин І.Б., Голубінка Ю.І. Методика визначення ексцентриситету фазового центра антени GPS-приймача // Збірник наукових праць VII-го міжнародного науково-технічного симпозіуму "Геоінформаційний моніторинг навколишнього середовища- GPS I GIS- технології", -Алушта, -2002, -С.54-57.
14. Третяк К.Р., Романишин І.Б., Голубінка Ю.І. До питання визначення ексцентриситета фазового центра антени GPS-приймача // Науково-технічний журнал "Геодезія, картографія та аерофотознімання" №62,-Львів, -2002, -С.87-96.

УДК 514.18:678.5.059:535.024:620.168:678.02:678.5.059
Колосова О.П., ас.
Національний технічний університет України «Київський політехнічний інститут»

ГЕОМЕТРИЧНЕ ТА МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ ПРОЦЕСУ ПРОСОЧЕННЯ ОРІЄНТОВАНИХ ВОЛОКНИСТИХ НАПОВНЮВАЧІВ РІДКИМИ ПОЛІМЕРНИМИ ЗВ'ЯЗУЮЧИМИ

Колосова О.П. Геометричне та математичне моделювання процесу просочення орієнтованих волокнистих наповнювачів рідкими полімерними зв'язуючими. Роглянуто приклади практичної адаптації методології геометричного і математичного моделювання для детермінації параметрів технологічного процесу просочення орієнтованих волокнистих наповнювачів розчинами полімерних зв'язуючих. Визначено комплекс структурних характеристик геометричної моделі орієнтованих волокнистих полімерних композитів, а саме пористість, питому внутрішню поверхню та ефективний (гідравлічний) капілярний радіус. Отримано задовільне співпадання теоретичних і практичних результатів досліджень. На основі мікроструктурного аналізу шліфів якісно встановлена ефективність застосування ультразвукової обробки при одержанні композитів. Досліджено, що в обробленому низькочастотним ультразвуком затверділому композиті практично відсутні сторонні включення між волокнами, а самі волокна розподілені практично регулярно. Отримані результати дають можливість оптимізувати кінетичні параметри процесу просочення та конструктивні параметри просочувального обладнання.
Ключові слова: геометричне моделювання, математичне моделювання, модель, структура, прогнозування, процес, технологія, просочення, параметр, полімер, зв'язуюче, волокно, композит

Колосова Е.П. Геометрическое и математическое моделирование процесса пропитки ориентированных волокнистых наполнителей жидкими полимерными связующими. Рассмотрены примеры практической адаптации методологии геометрического и математического моделирования для детерминации параметров технологического процесса пропитки ориентированных волокнистых наполнителей растворами полимерных связующих. Определен комплекс структурных характеристик геометрической модели ориентированных волокнистых полимерных композитов, а именно пористость, удельная внутренняя поверхность и эффективный (гидравлический) капиллярный радиус. Получено удовлетворительное совпадение теоретических и практических результатов исследований. На основе микроструктурного анализа шлифов качественно установлена эффективность применения ультразвуковой обработки при получении композитов. Установлено, что в обработанном низкочастотным ультразвуком затвердевшем композите практически отсутствуют посторонние включения между волокнами, а сами волокна распределены практически регулярно. Полученные результаты дают возможность оптимизировать кинетические параметры процесса пропитки и конструктивные параметры пропиточного оборудования.
Ключевые слова: геометрическое моделирование, математическое моделирование, модель, структура, прогнозирования, процесс, технология, пропитки, параметр, полимер, связующее, волокно, композит

Kolosova E.P. Geometric and mathematical modeling of the impregnation process of oriented fibrous fillers by liquid polymeric binders. The relevance and practical examples of adaptation of the methodology of geometric and mathematical modeling for determination of process parameters of impregnation of oriented fibrous fillers by solutions of polymeric binders is considered. The complex of the structural characteristics of the geometric model of oriented fibrous polymeric composites, namely porosity and specific internal surface of the effective (hydraulic) capillary radius is determined. A satisfactory agreement between the theoretical and practical research results is obtained. On the basis of microstructural analysis of thin sections is qualitatively established the effectiveness of ultrasonic treatment for formation of composites. It is found that in low frequency ultrasonic treated hardened composite there is almost no foreign inclusions between the fibers, and the fibers themselves are distributed substantially regularly. The results obtained make it possible to optimize the kinetic parameters of the impregnation process and design parameters of the impregnation equipment.
Keywords: geometric modeling, mathematical modeling, model, structure, prediction, process, technology impregnation, parameter, polymer, binder, fiber, composite.

Постановка проблеми. Просочувально-сушильне обладнання широко застосовується при виготовленні реактопластичних армованих полімерних композиційних матеріалів, і, зокрема, препрегів [1□4]. Оскільки основними структурними елементами будь-якого виду вищезазначеного обладнання є вузли для просочування і наступного сушіння, виникає необхідність прогнозування параметрів технологічного процесу просочення. Останній в основному зумовлює продуктивність просочувальних ліній, а також якість полімерної продукції, що одержується.

Просочування здійснюють для надання висушеним матеріалам, що просочилися, певних властивостей (міцності, волого- і вогнестійкості, пружності, забарвлення і т.п.). Власне процес просочування включає в себе наступні основні стадії [1□4]: нанесення просочувального складу на поверхню армуючого волокнистого наповнювача (ВН) чи занурення ВН у просочувальну ванну; проникнення просочувального складу в макропори ВН; дифузію просочувального складу до поверхні волокон; дифузію просочувального складу всередину волокон.

Проте вказані стадії не мають чітких кордонів, оскільки реальні ВН володіють неоднорідною структурою. До того ж на різних дільницях вищевказані стадії просочування протікають з різною швидкістю, що нерідко приводить до їх поєднання.

Не зважаючи на досягнуті успіхи у формуванні, переробці, а також одержанні армованих композитів, конструювання намотувальних виробів на базі орієнтованих і рулонних ВН і полімерних зв'язуючих (ПЗ), а також процеси та обладнання для їх виробництва до цих пір у більшості випадків базуються на евристичних началах і особистому досвіді розробників. Це призводить до нераціональних, і тим більше до нерентабельних конструкцій та виробів з таких композитів.

Виходячи з вищенаведеного, використання геометричного та математичного моделювання при проектуванні технологічного процесу просочування уявляється головною ланкою в підвищенні продуктивності проектування й техніко-економічних показників будь-якого виду обладнання цього типу загалом. Тому необхідно дослідити цю проблему більш детально.

Аналіз досліджень і публікацій. У загальному випадку структуру армованого полімерного композиту уявляють як капілярно-пористе середовище (тіло) [5]. Рушійними силами процесу просочування є саме капілярне всмоктування [5], а також сили, що впливають на розчин ПЗ і прискорюють його рух усередину ВН, або орієнтованого чи односпрямованого волокнистого наповнювача (ОВН).

Цим силам перешкоджають опір течії розчину ПЗ у порах ВН і поверхневе натягнення розчину ПЗ. Оскільки просочування ВН розчином ПЗ можна розглядати як витіснення повітря з пор і капілярів ВН та заміну його розчином ПЗ [1], цей процес можна класифікувати як один з прикладів капілярних явищ [5].

Тривалість перебування ВН в просочувальному розчині (час просочування) визначається швидкістю руху полотна і розмірами просочувальної ємності (ванни). У роботі [1] пропонується опис процесу просочування ВН рідким ПЗ проводити за допомогою відомого рівняння Дарсі (1):

$$V = \sqrt{S\left(\frac{2k_n\varepsilon_н}{\eta}\right)\frac{P}{t}}, \qquad (1)$$

де V – швидкість просочування (або збільшення вмісту ПЗ); S – площа поверхні ВН, що просочується; k_n – експериментальна константа; $\varepsilon_н$ – пористість ВН; η – в'язкість розчину ПЗ; P – тиск при просочуванні; t – час просочування.

Рівняння (1), зокрема, показує, що для збільшення вмісту ПЗ (або швидкості V) у структурі ВН вдвічі необхідно в чотири рази збільшити час перебування ВН у розчині ПЗ. У той же час уміст ПЗ (або швидкість V) є обернено пропорційним в'язкості розчину ПЗ, яка, як відомо, залежить від температури просочування [1].

Відомо [6□8], що базовим питанням при детермінуванні параметрів кінетичного рівняння процесу просочування є коректне знаходження характеристик геометричної (фізичної) моделі структури ОВН, на основі якої отримують це рівняння, а саме пористості ε, питомої внутрішньої поверхні $S_{пит}$ та ефективного (гідравлічного) капілярного радіусу $r_{еф}$ того чи іншого типу використовуваного для просочування типу ОВН.

Було досліджено, що при побудові геометричної (фізичної) моделі структури ОВН є доцільним використання структурного підходу (а саме мікроструктурного аналізу перетину композиту на основі ОВН) для визначення шуканих параметрів, зокрема, ефективного (гідравлічного) капілярного радіусу $r_{еф}$ [8].

При цьому найбільш адекватним уявленням структури ОВН є структура капілярно-пористого тіла. Остання складається із системи паралельно-звивистих капілярів різних радіусів, яку для випадкової (стохастичної) величини їх розміщення можна адекватно описати за допомогою функції розподілу пор радіусу ρ по розмірах $\varphi(\rho)$ [5].

Відповідно до використовуваного підходу в роботі [8], після затвердіння просоченого ОВН за визначеного зусилля натягнення ОВН при просочуванні роблять мікрошліф його перетину у поперечному до волокон напрямі. Далі у перетині мікрошліфу експериментально досліджують розподіл довжин екстхорд волокон (відстаней між суміжними волокнами), або функцію розподілу $\varphi(\rho) = G(\ell)$ випадкових пор за розмірами ℓ. При цьому екстхорди являють у нашому випадку

еквівалентний (ефективний) діаметр (чи подвійний радіус $r_{еф}$) пор, причому довжини екстхорд носять випадковий характер.

Детермінацію теоретичної кривої розподілу $\varphi(\rho)= G(\ell)$ проводять за умови найкращого наближення (тобто мінімального відхилення) теоретичної кривої розподілу до експериментальних ординат. Надалі за допомогою знайденої функції розподілу довжин екстхорд $\varphi(\rho) = G(\ell)$ обчислюють пористість ε, питому внутрішню поверхню $S_{пит}$ та ефективний (або еквівалентний) капілярний радіус $r_{еф}$ шуканого ОВН як капілярно-пористого тіла.

Цілі статті. Метою даної роботи є застосування методології геометричного та математичного моделювання для прогнозування параметрів технологічного процесу просочення орієнтованих волокнистих наповнювачів рідкими полімерними зв'язуючими, що дозволить оптимізувати кінетичні параметри цього процесу та конструктивні параметри просочувального обладнання.

Основні матеріали дослідження. У роботі [10] була розглянута геометрична модель середовища орієнтованого волокнистого полімерного композиту (ОВПК) у вигляді системи циліндричних волокон, осі яких у загальному випадку паралельні і розташовані у вузлах подвійноперіодичної (регулярної) решітки паралелограмів, а простір між волокнами заповнений затверділим ПЗ. Визначення параметрів цієї геометричної моделі здійснюється на базі використання методології теорії інтегральної геометрії і геометричних імовірностей шляхом дослідження (функції) розподілу відстаней між суміжними колами, що розташовані у вузлах подвійноперіодичної решітки, і які перетинаються випадковими січними.

Здійснимо детермінацію структурних параметрів цієї геометричної моделі ОВПК за припущення незмінності розташування волокон у структурі просоченого ОВН під час його сушіння за стаціонарності значення зусилля натягнення ОВН. Пористість ε у капілярно-пористому тілі згідно [5], наприклад, визначається так:

$$\varepsilon = N^0 \pi \beta_i \int_0^\infty \rho^2 \varphi(\rho) d\rho = \frac{N^0 \pi \beta_o \bar{\rho}^2}{4} = \frac{N^0 \pi \beta_o}{4} \int_0^\infty \rho^2 \varphi(\rho) d\rho, \tag{1}$$

де N^o — кількість замірів поміж волокнами у площині шліфу композиту.

У свою чергу, питому внутрішню поверхню $S_{пит}$ можна визначити таким чином [5]:

$$S_{пит} = k_o N^o \pi \beta \int_0^\infty \rho \varphi(\rho) d\rho = \kappa_o N^o \pi \beta_o \bar{\rho}, \tag{2}$$

де $\bar{\rho}$ — середній радіус пор, а коефіцієнти κ_o і β_o можна трактувати як поправочні коефіцієнти, що враховують специфіку структури ОВН як капілярно-пористого середовища (і які означають відповідно шорсткуватість поверхні і звивистість еквівалентного циліндричного капіляра).

Якщо прийняти відповідно до робіт [6, 7] як ефективний (або еквівалентний капілярний радіус) $r_{еф}$ шуканого ОВН гідравлічний радіус, що визначається за аналогією з «ідеальним» циліндричним капіляром як відношення подвоєного об'єму пор до їх поверхні, то будемо мати:

$$r_{еф} = \frac{2\varepsilon}{S_{пит}} = \frac{2}{\kappa_o} \left[\int_0^\infty \rho^2 \varphi(\rho) d\rho \Big/ \int_0^\infty \rho \varphi(\rho) d\rho \right]. \tag{3}$$

Було, зокрема, встановлено [8], що для практичних обчислень функцію розподілу довжин екстхорд волокон, яка мінімізує відхилення від експериментальних ординат, зручно описувати у такому вигляді:

$$G(\lambda) = \varphi(\rho) = \frac{c_к b^{(1/c_к)}}{\Gamma(1/c_к)} e^{-\left(b_к \rho^{c_к}\right)}, \tag{4}$$

де $b_к$, $c_к$ — позитивні константи, що визначаються за умови найкращого наближення теоретичної кривої (4) до експериментальних ординат; Г – гамма-функція Ейлера.

Для цього випадку рівняння (4) наведемо й інші залежності. Так, зокрема, математичне очікування випадкової величини ρ, функція розподілу $\varphi(\rho)$ якої описується за допомогою рівняння (4), має такий вигляд:

$$M(\rho)=\int_0^{\infty}\rho\varphi(\rho)d\rho=\frac{\Gamma(2/c_к)}{b^{(1/c_к)}\cdot\Gamma(1/c_к)}. \tag{5}$$

Формула (1) для пористості ε прийме наступний вигляд:

$$\varepsilon=\frac{N^0\pi\beta_o}{4}\int_0^{\infty}\rho^2\varphi(\rho)d\rho=\frac{N^0\pi\beta_o}{4}\cdot\frac{\Gamma(3/c_к)}{b^{(2/c_к)}\cdot\Gamma(1/c_к)}, \tag{6}$$

а формула для еквівалентного капілярного радіусу (3) запишеться таким чином:

$$r_{еф}=\frac{2}{\kappa_o}\left[\int_0^{\infty}\rho^2\varphi(\rho)d\rho/\int_0^{\infty}\rho\varphi(\rho)d\rho\right]=\sqrt{2}\,\frac{\Gamma(1/c_к)}{b^{(1/c_к)}\cdot\Gamma(2/c_к)}. \tag{7}$$

Рівняння (3), (4), (6) і (7) повністю визначають комплекс структурних характеристик геометричної моделі ОВПК, а саме пористість ε, питому внутрішню поверхню $S_{пит}$ та ефективний (гідравлічний) капілярний радіус $r_{еф}$.

Прогнозування кінетичних параметрів поздовжнього просочування орієнтованих волокнистих наповнювачів рідкими полімерними зв'язуючими

В роботі [11] було наведено прогностичне кінетичне рівняння процесу поздовжнього просочення ОВН рідкими ПЗ:

$$t_1=\frac{\eta S_{тр}S_{пит}\sigma\cos\theta}{\varepsilon^2\gamma^2g^2r_{еф}}\left[\ln\left|\frac{\exp(1)}{1-\varepsilon\gamma gh/S_{уд}\sigma\cos\theta}\right|-\frac{\varepsilon\gamma gh}{S_{уд}\sigma\cos\theta}\right], \tag{8}$$

де t_1 – час поздовжнього просочування; h – висота поздовжнього просочування; $\sigma\cos\theta$ – змочувальна здатність; θ – крайовий кут змочування; η – динамічна в'язкість розчину ПЗ; γ – динамічна в'язкість розчину ПЗ.

Рівняння (8) після підстановки відповідних величин з урахуванням рівнянь (2), (6) і (7) прийме такий вигляд:

$$t_1=\frac{\eta S_{тр}S_{пит}\sigma\cos\theta}{\varepsilon^2\gamma^2g^2}\cdot\frac{b^{(1/c_к)}\cdot\Gamma(2/c_к)}{\sqrt{2}\cdot\Gamma(3/c_к)}\left[\ln\left|\frac{\exp(1)}{S_{пит}\sigma\cos\theta}\right|+\frac{1}{2}\cdot\left(\frac{\varepsilon\gamma gh}{S_{пит}\sigma\cos\theta}\right)^2\right]. \tag{9}$$

Експериментальну перевірку прогностичного кінетичного рівняння (9) здійснювали на прикладі скловолокнистого і органоволокнистого джгутів (діаметром 2,5 мм і довжиною 100 мм), які просочували розчином епоксидної смоли марки ЕД-20 при температурі 50 °С і зусиллі натягнення джгута 30 Н/м. Виміряна реовіскозіметром динамічна в'язкість розчином епоксидної смоли при даній температурі склала η = 0,48 Па·с, а змочувальна здатність $\sigma\cos\theta$, обрахована по максимальній висоті підйому зв'язуючого по волокну під дією сил поверхневого натягнення за формулою [12]:

$$\sigma\cos\theta=\frac{h_\infty\gamma gR_к}{2}, \tag{10}$$

де h_∞ – максимальна висота підйому рідини (поздовжнього просочування); R_k – радіус капіляра, склала $\sigma\cos\theta$ = $2\cdot10^{-2}$ Н/м.

З рис. 1, на якому показана кінетична крива поздовжнього просочення, видно добрий збіг експериментальних і розрахованих за рівнянням (9) значень.

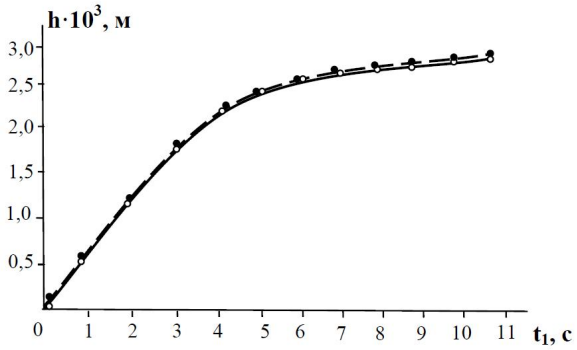
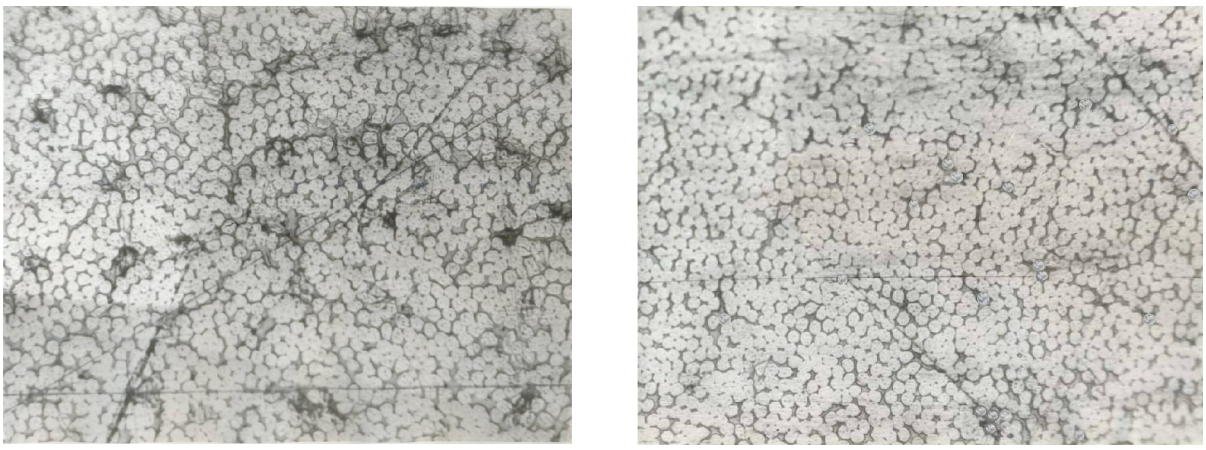


Рис. 1. Кінетичні криві поздовжнього просочування скловолокнистого джгута розчином епоксидного звязуючого ЕД-20 при температурі 50 °С і зусиллі натягнення джгута 30 Н/м: (●) – прогностична крива, побудована за рівнянням (9); (○) – експериментальні значення.

Таким чином, розроблений удосконалений підхід дає можливість здійснювати моделювання параметрів технологічного процесу просочення ОВН рідкими ПЗ з врахуванням інтегральних характеристик ОВН як капілярно-пористого тіла.

Порівняльний аналіз структури орієнтованих волокнистих наповнювачів, просочених рідкими епоксидними зв'язуючими, за ультразвукової дії

На рис. 2 наведена типова фотографія мікрошліфу поперечного перерізу орієнтованого епоксидного органопластика, одержаного за «вільного» просочення, тобто без ультразвукової (УЗ) дії, так з УЗ-дією.



а б

Рис. 2. Типова фотографія мікрошліфу поперечного перерізу орієнтованого епоксидного органопластика, одержаного без УЗ-обробки (*а*) і за ефективних режимів УЗ-обробки (*б*). Збільшення х1200

Проведені дослідження [13] свідчать про покращення експлуатаційних властивостей затверділої полімерної матриці. Внаслідок цього можна очікувати й на поліпшення експлуатаційних властивостей ОВПК на її основі, про що опосередковано може свідчити розподіл волокон у структурі шліфу (див. рис. 2, *б*).

Порівнюючи якісно обидва шліфи (рис. 2, *а,б*), можна помітити, що в обробленому низькочастотним УЗ затверділому ОВПК практично відсутні сторонні включення (у т.ч. повітряні порожнини) між волокнами, а самі волокна розподілені практично регулярно. Це свідчить на користь застосування апроксимуючої подвійно-періодичної структурної моделі ОВПК для прогнозування технологічних параметрів процесу просочення (див. рис. 2, *а,б*), а також про ефективність застосування УЗ-обробки при одержанні ОВПК [14-15].

На рис. 3 показана експериментальна гістограма, а також теоретична крива розподілу довжин екстхорд суміжних волокон (кіл) (4) $f_{\Delta}(z) = G(\ell)$ в структурі ОПВК.

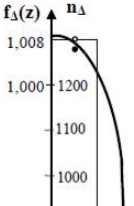


Рис. 3. Гістограма і криві розподілу довжин екстхорд суміжних волокон (кіл) f_Δ (z), отриманих за ефективних режимів низькочастотної ультразвукової дії: експериментальні ординати (○); теоретичний розподіл (4) (●); n_Δ □ кількість замірів.

Наведемо значення виразів (5) – (7) для функції розподілу довжин екстхорд волокон, яка мінімізує відхилення від експериментальних ординат (○) – див. рис. 3. Так, при кількості замірів поміж волокнами N^o = 2500 у площині шліфу композиту (див. рис. 2, *а, б*) і значеннях β_o = 1,1, $\kappa_o \approx \sqrt{2}$ [6, 9] математичне очікування (5) дорівнює $M(\rho)$ = 8,158 мкм. При цьому постійні рівняння (4) для необробленого УЗ ОВПК становлять: $b_к$ = 0,129, $c_к$ = 0,9812, пористість (6) ε = 0,28 (тобто коефіцієнт армування або об'ємний вміст ОВН у структурі композиту складає $\xi_a = 1 - \varepsilon = 0{,}72$), а ефективний (еквівалентний) капілярний радіус (7) $r_{еф} = \overline{\rho}$ = 5,96 мкм ≈ 6 мкм.

Знайдені для ОВПК, обробленого УЗ, постійні теоретичного рівняння (4) становлять: $b_к$ = 0,125, $c_к$ = 0,992, пористість (6) ε = 0,27 (тобто коефіцієнт армування чи об'ємний вміст ОВН у структурі композиту складає $\xi_a = 1 - \varepsilon =$ 0,73), а ефективний (еквівалентний) капілярний радіус (7) $r_{еф} = \overline{\rho}$ = 5,77 мкм ≈ 5,8 мкм.

За отриманими експериментальним ординатами (○) гістограми розподілу довжин екстхорд суміжних волокон проводиться теоретична крива (4) з урахуванням її найменшого відхилення від експериментальних ординат (○). Надалі здійснюють оптимізацію кінетичних параметрів процесу просочення (час, висота чи швидкість просочування) та конструктивні параметри (зокрема, габарити ванни) просочувального обладнання.

Висновки. Роглянуто приклад практичної адаптації методології геометричного і математичного моделювання для детермінації параметрів технологічного процесу просочення орієнтованих волокнистих наповнювачів розчинами полімерних зв'язуючих. Отримано задовільне співпадання теоретичних і практичних результатів досліджень. Також на основі мікроструктурного аналізу шліфів якісно встановлена ефективність застосування ультразвукової обробки при одержанні композитів. Отримані результати дають можливість оптимізувати кінетичні параметри процесу просочення та конструктивні параметри просочувального обладнання.

1. Шалун Г. Б. Слоистые пластики / Г.Б. Шалун, Е. М. Сурженко. – Л.: Химия, 1978. – 232 с.
2. Цыплаков О. Г. Научные основы технологии композиционно-волокнистых материалов. Ч.1 / Цыплаков О. Г. – Пермь, 1974. – 317с.
3. Плоткин Л. Г. Технология и оборудование пропитки бумаги полимерами / Л. Г. Плоткин, Г. В. Шалун – М.: Лесная промышленность, 1985. – 119 с.
4. Коновалов В. И. Пропиточно-сушильное и клеепромазочное оборудование / Коновалов В. И. □ М.: Химия, 1989. – 224 с.
5. Аксельруд Г. А. Введение в капиллярно-химическую технологию / Г. А. Аксельруд, М. А. Альтшулер. – М.: Химия, 1983. – 264 с.
6. Чизмаджев Ю. А. Макрокинетика процессов в пористых средах / Ю. А. Чизмаджев, В. С. Маркин, М. Р. Тарасевич и

др. – М.: Наука, 1971. – 364 с.
7. Хейфец Л.И., Неймарк А.В. Многофазные процессы в пористых средах / Л.И. Хейфец, А.В.Неймарк. – М.: Химия, 1982. – 320 с.
8. Колосов О.Є. До вибору фізичної моделі капілярно-пористого середовища на основі орієнтованих волокнистих наповнювачів / О.Є.Колосов // Вісник НТУУ КПІ. Сер. «Машинобудування». – 2010. □ №59. □ С. 96 □ 101.
9. Цыплаков О. Г. Научные основы технологии композиционно-волокнистых материалов. Ч.1 / Цыплаков О. Г. – Пермь, 1974. □ 317с.
10. 10.Колосов О.Є. Фізична модель структури капілярно-пористого середовища на основі орієнтованих волокнистих наповнювачів / О.Є. Колосов, В.І. Сівецький, Л.А. Кричковска, О.П. Колосова // Восточно-Европейский журнал передовых технологий. – 2013. – №1. – С. 6□9.
11. Колосов О.Є. Кінетика процесу просочування волокнистих наповнювачів композиціями епоксидних полімерів / О.Є. Колосов, В.І. Сівецький, Л.А. Кричковска, О.П. Колосова // Восточно-Европейский журнал передовых технологий. – 2013. – №2. – С. 13□16.
12. Малкин А.Я. Диффузия и вязкость полимеров. Методы измерения / А.Я.Малкин, А.Е.Чалых. – М.: Химия, 1979. – 304 с.
13. Колосов О.Є. Використання методів математичного та експериментально-статистичного моделювання для оптимізації технологічних параметрів ультразвукового одержання полімерних композиційних матеріалів / О.Є. Колосов, В.І. Сівецький, В.С. Кривошеєв, О.П. Колосова // Журнал Кам.-Под. нац. ун-ту «Математичне та комп'ютерне моделювання. Сер.: Техн. науки». – 2014. – Вип.11. – С. 61 – 72.
14. Колосов О.Є. Математичне моделювання базових процесів виготовлення полімерних композиційних матеріалів із застосуванням ультразвукової модифікації / О.Є. Колосов, В.І. Сівецький, Є.М. Панов та ін. – К.: ВД «Едельвейс», 2012. □ 268 с.
15. Колосов О.Є. Одержання волокнистонаповнених реактопластичних полімерних композиційних матеріалів із застосуванням ультразвуку / О.Є.Колосов, В.І. Сівецький, О.П. Колосова. – К.: ВПК «Політехніка», 2015. – 295 с.

УДК 004.023
Марченко О.І. к.т.н, доцент, Хоптинець В.А. магістрант
Національний технічний університет України «Київський політехнічний інститут»

ТРАНСЛЯЦІЯ ПРОГРАМ З ПРОЦЕДУРНИХ МОВ ПРОГРАМУВАННЯ У ФУНКЦІОНАЛЬНІ МОВИ З ВИКОРИСТАННЯМ ГРАФУ ЗАЛЕЖНОСТІ ДАНИХ

Марченко О.І., Хоптинець В.А. Трансляція програм з процедурних мов програмування у функціональні мови з використанням графу залежності даних. У статті пропонується спосіб трансляції програм з процедурних мов програмування у функціональні мови з використанням внутрішньої форми, що відповідає функціональній парадигмі програмування – графу залежності даних. На відміну від інших внутрішніх форм, таких як граф потоку виконання, граф залежностей даних, цей граф явно виражає залежність по даним між операторами вхідної мови, що дає змогу виділили обчислення без сторонніх ефектів.
Ключові слова: трансляція, процедурне програмування, функціональне програмування, внутрішня форма, граф залежності даних.

Марченко А.И., Хоптынец В.А. Трансляция программ с процедурных языков программирования в функциональные языки с использованием графа зависимостей данных. В статье предлагается способ трансляции программ с процедурных языков программирования в функциональные языки с использованием внутренней формы, которая соответствует функциональной парадигме программирования – графа зависимостей данных. В отличии от других внутренних форм, таких как граф потока выполнения, граф зависимостей данных явно выражает зависимости по данным между операторами исходного языка, что дает возможность отделить вычисления без побочных эффектов.
Ключевые слова: трансляция, процедурное программирование, функциональное программирование, внутренняя форма, граф зависимостей данных.

Marchenko O.I., Khoptynec V.A. Translation of programs from procedural languages to functional languages using value dependence graph. Technique for programs translation from procedural programming languages to functional languages using functional-oriented intermediate representation – value dependence graph, is proposed in this paper. In contrast to other intermediate representations like control flow graph, value dependence graph exposes dependencies among statements of source languages, that helps to extract evaluations without side effects.
Keywords: translation, procedural programming, functional programming, intermediate representation, value dependence graph.

Вступ. Починаючи з середини XX сторіччя було створено велику кількість програмного забезпечення (ПЗ), яке досі, незважаючи на еволюцію апаратного забезпечення, актуальне і потребує підтримки, оскільки на ньому тримаються критичні системи тих чи інших підприємств. Підтримка таких систем тягне за собою певні додаткові витрати (порівняно із сучасним ПЗ), що пов'язані з використанням застарілих програмних інструментів. Тому, актуальною задачею є автоматична трансляція ПЗ на рівні вихідного коду з одних мов високого рівня у інші. В даному випадку, з мов, що практично вийшли з використання, у більш сучасні мови.

Зараз багато спеціалістів відзначають перспективність декларативної функціональної парадигми для створення нових великих програмних систем. Функціональна парадигма надає ряд переваг з точки зору розпаралелення програм та автоматичної верифікації.

Оскільки застарілі мови програмування високого рівня, як правило, є процедурними, то дослідження способів ефективної трансляції програм з процедурних мов програмування у функціональні мови є перспективним напрямом. Саме дослідженню таких способів трансляції присвячена дана робота.

Постановка наукової проблеми. Метою даної роботи є аналіз існуючих способів трансляції програм з процедурних мов програмування у функціональні з метою отримання способу, що забезпечує автоматичне відокремлення обчислень, що не містять сторонніх ефектів, від обчислень в процедурній парадигмі.

Аналіз існуючих способів.

Процедурне програмування – парадигма програмування, при якій порядок обробки операторів комп'ютером є строго визначеним.

Функціональне програмування – парадигма програмування, в якій процес програмування трактується як обчислення значень функцій в математичному сенсі, і порядок обчислення функцій визначається наявністю даних для їх обчислення, а не порядком їх розташування у тексті програми.

При використанні обох парадигм програмування, як процедурної, так і функціональної, складовими частинами програми є набір операторів або виразів у вигляді підпрограм з певними вхідними параметрами. У випадку процедурного програмування такі підпрограми є або процедурами, або функціями, а у випадку функціонального програмування – функціями. Різниця між функціями процедурного і функціонального стилю полягає у семантиці. У першому випадку функція є підпрограмою, яка може приймати якісь параметри та повертати значення. У функціональному програмуванні, функція має таке ж значення, як і в математиці. Це призводить до того, що функції у функціональних мовах програмування мають бути «чистими», тобто не містити сторонніх ефектів.

Найпростіший спосіб – пряма трансляція. У цьому випадку, вхідна програма транслюється літерально, моделюючи нехарактерні для вихідної мови абстракції вхідної мови.

Сучасні мови програмування, що підтримують як процедурну, так і функціональну парадигми, наприклад Common Lisp, дозволяють використовувати сторонні ефекти, зокрема оператор присвоювання замінюється на виклик макросу setf. Також, в Common Lisp можна використовувати динамічні змінні, які легко замінюють глобальні змінні. Таким чином, трансляція в Common Lisp є достатньо прямою. Приклад програми мовою C показано на рис.1. Відповідна програма мовою Common Lisp – на рис.2.

```
int global_foo;

int foo(int a) {
  int x = 123 + a;
  if (x > global_foo) {
    global_foo = x;
  }
  else {
    global_foo = a;
  }
  return global_foo + a + x;
}
```

Рис.1. Програма мовою C

```
(defvar *global_foo*)

(defun foo (a)
  (let ((x (+ 123 a)))
    (if (> x *global_foo*)
      (setf *global_foo* x)
      (setf *global_foo* a))
   (+ *global_foo* a x))
```

Рис.2. Програма мовою Common Lisp

Натомість в чисто функціональних мовах програмування, таких як Haskell, не можна просто використовувати оператор присвоювання і глобальні змінні. Для емуляції такої семантики у Haskell доводиться використовувати спеціальний тип IORef у зв'язці із монадою вводу-виводу. Попередній приклад мовою Haskell наведено на рис.3.

Отже, можна зробити висновок, що проста пряма трансляція має сенс лише у мовах, які крім основної функціональної парадигми, мають також засоби підтримки процедурної парадигми, бо в іншому випадку результуючий код стає занадто ускладненим. Проблема з цим підходом полягає у тому, що порушуються абстракції, на які спирається функціональна парадигма. Це призводить до ускладнення супроводу таких програм, а також перешкоджає компілятору у виконанні додаткових, більш ефективних, способів оптимізації коду.

```
global_foo :: IORef Int
global_foo = unsafePerformIO $ newIORef
undefined

foo :: Int -> IO Int
foo a = do
  let x = 123 + a
  _global_foo <- readIORef global_foo
  if x > _global_foo then
    writeIORef global_foo x
  else
    writeIOref global_foo a
  _global_foo_return <- readIORef
global_foo
```

Рис. 3. Програма мовою Haskell

Іншим підходом до трансляції процедурних мов у функціональні є використання мово-незалежних внутрішніх (проміжних) форм. Внутрішні форми відрізняються між собою інформацією про вхідну програму, яку вони використовують. Найпростіша внутрішня форма – дерево розбору, яке представляє структуру програми. Для виконання аналізу, оптимізації та трансформації вхідної програми у вихідну програму доцільно попередньо перевести дерево розбору у більш придатну для цього внутрішню форму.

Поширеною внутрішньою формою є граф потоку виконання (CFG) у зв'язці з формою одиничного присвоювання (SSA), на базі якого можна виконувати аналізи потоку даних (DFA). До недоліків цієї форми відноситься сильна впорядкованість операторів, а також відсутність інформації про залежності між операторами. Приклад графу потоку виконання для попередньої програми наведено на рис.4.

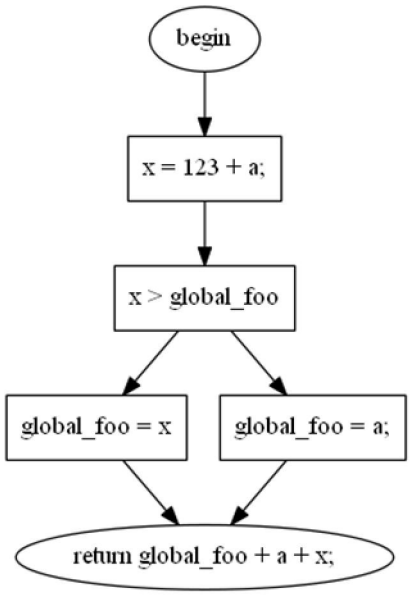


Рис. 4. Граф потоку виконання

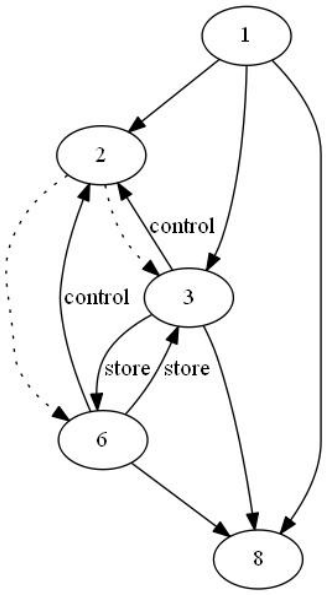


Рис. 5. Граф залежностей

Альтернативою CFG є граф залежностей (PDG) [1]. В цьому графі вершинами також є оператори, а дуги - залежності між ними. Така форма дозволяє «послабити» впорядкованість, роблячи її неявною, що дозволяє виконувати деякі специфічні трансформації, а також векторизуючі оптимізації. Проблема PDG полягає в тому, що структурною одиницею є оператор,

зокрема, оператор присвоювання. Тобто, така форма добре підходить для подальшої генерації асемблерного коду або коду на процедурній мові високого рівня. Граф залежностей для наведеного вище прикладу зображено на рис.5. Кожен вузол відповідає оператору вхідної програми (номер вузла відповідає номеру рядка у вхідній програмі). 1 – визначення змінної x; 2 – порівняння x та global_foo; 3 – запис x у global_foo; 6 – запис a у global_foo.

Виклад основного матеріалу й обґрунтування отриманих результатів дослідження.

Пропонується спосіб трансляції програм із процедурних мов програмування у функціональні мови, що базується на використанні графу залежності даних (VDG) як внутрішньої форми представлення програми [2]. Граф залежності даних дозволяє позбавитись від інформації про оператор, залишаючи натомість інформацію про залежність даних у цих операторах. У цьому графі вершинами є дані і операції над ними, а як дуги використовуються залежності. Таким чином, граф залежності даних добре відповідає функціональній парадигмі і тому придатна для ефективної генерації коду функціональної мови програмування. При такому підході компілятор зможе автоматично виявити чисто функціональні фрагменти коду, а для інших фрагментів використати монаду стану [3]. До недоліків цієї форми відноситься складність побудови графу залежності даних за програмою на вхідній процедурній мові. Граф залежності даних для наведеного вище прикладу зображено на рис. 6.

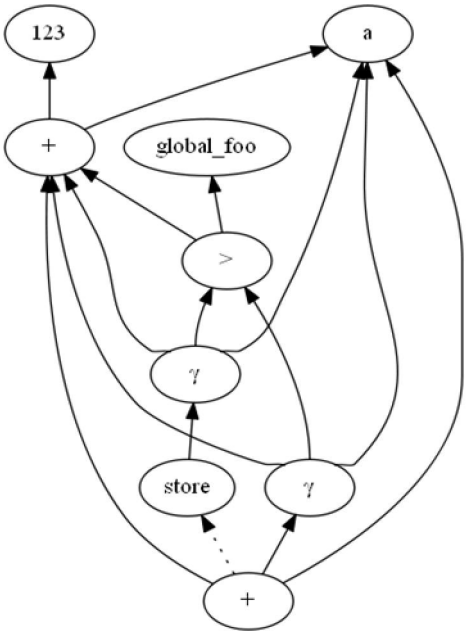


Рис. 6. Граф залежності даних

У цьому графі вершини відображають обчислення, а дуги – значення, від яких залежить обчислення. Якщо операція E1 залежить від результату обчислення E2, то граф містить дугу від E1 до E2. Вершини можуть бути кількох типів:

1. Примітивна операція. До цього типу відносяться арифметичні операції, константи тощо.

2. Булевий селектор (γ), що відповідає умовним виразам. Ці вершини видають одне з вхідних значень в залежності від умовного виразу.

3. Замикання (λ), що видає значення-функцію.

4. Виклик функції. Ці вершини залежать від значення-функції та параметрів (що відповідають формальним параметрам вхідної функції). Результатом цих вершин є значення із вершини результату вхідної функції.

5. Формальний параметр. Ці вершини не мають залежностей, але мають значення, що було передано у дану функцію через даний формальний параметр.

6. Вільна змінна. Такі вершини містять значення зі зовнішнього контексту.

7. Вершина результату. Від цих вершин інші вершини не залежать явно. Самі вершини залежать від значення, яке є значенням всієї функції. Будь-який граф залежності даних містить не менше однієї вершини результату (по одній вершині для кожного значення, що повертається).

8. Читання із глобального стану.

9. Запис у глобальний стан.

Побудова графу залежності даних із дерева розбору складається зі п'яти основних етапів. По-перше, необхідно розбити дерево на базові блоки і побудувати граф потоку виконання.

На другому етапі необхідно побудувати граф залежності записів (store dependency graph – SDG). Цей граф замість інформації про залежності у виконанні відображає потік зміни значення запису. Запис містить інформацію про значення усіх змінних. При обчисленні нового значення певної змінної використовуються поточні значенні запису. Вузлам розгалуження у графі потоку виконання відповідає розгалуження запису з подальшим вибором за допомогою γ-вузла. Тіла циклів замінюються функція, а самі цикли на виклики цих функцій.

Наступним кроком необхідно виконати вбудовування (inlining) нерекурсивних викликів. Після чого можна виконувати символьну інтерпретацію графу залежності записів. При цьому створюються вершини, що відповідають операціям, що виконуються над значеннями. Під час інтерпретації використовуються таблиці пошуку, в яких міститься інформація з якої вершини можна отримати поточне значення певної змінної. Якщо в таблиці відсутній вказівник на відповідну вершину (наприклад, значення було неявно змінено у функції), то використовуються спеціальні вершини читання із глобального стану.

На цьому етапі граф залежності даних вважається створеним, але для подальшої генерації варто виконати аналіз циклів та видалення зайвих операцій запису і читання.

Варто зазначити, що в результаті побудови графу залежності даних, над вхідною програмою автоматично виконуються нумерація значень і розповсюдження копій та констант.

Висновки

Проаналізовані способи трансляції програм з процедурних мов програмування у функціональні можуть бути закладені в основу відповідних спеціалізованих трансляторів. Як спосіб прямої трансляції, так і спосіб, що базуються на використанні внутрішніх форм, мають свої переваги і недоліки.

Спосіб прямої трансляції є простішим у реалізації, але має обмежене застосування. Його можна використати лише тоді, коли вихідна мова підтримує імперативну парадигму, або коли вхідна програма не містить сторонніх ефектів, тобто семантично є функціонально чистою, що майже не зустрічається на практиці.

Спосіб, що базується на використанні графу залежності даних є більш складним у реалізації, але він надає змогу автоматично виділити функціонально чисті фрагменти коду від наявних імперативних фрагментів, відокремивши їх за допомогою монади стану. Крім того, у порівнянні з іншими внутрішніми формами, граф залежності даних дозволяє ефективніше виконувати деякі подальші оптимізації, таких як slicing [4].

Напрямом подальшого дослідження є розробка комбінованих способів трансляції, що поєднують описані вище способи трансляції.

1. Jeanne Ferrante, Karl J. Ottenstein, Joe D. Warren "The Program Dependence Graph and Its Use in Optimization" - ACM Transactions on Programming Languages and Systems, том 9, №3, Липень 1987, с. 319-349.
2. Daniel Weise, Roger F. Crew, Michael Ernst, Bjarne Steensgaard "Value Dependence Graphs: Representation Without Taxation", Principles Of Programming Languages 1994.
3. Philip Wadler "Monads for functional programming" – University of Glasgow G12 8QQ
4. Mark Weiser. "Program slicing". Proceedings of the 5th International Conference on Software Engineering, pages 439–449, IEEE Computer Society Press, March 1981.

УДК 004.023
Марченко О.І. к.т.н. доцент, Щербина Б.О. магістрант
Національний технічний університет України «Київський політехнічний інститут»

СПОСІБ ТРАНСЛЯЦІЇ ТИПІВ ДАНИХ МОВИ COBOL В ТИПИ ДАНИХ СУЧАСНИХ МОВ ПРОГРАМУВАННЯ

Марченко О.І., Щербина Б.О. Спосіб трансляції типів даних мови COBOL в типи даних сучасних мов програмування. У статті запропоновано модифікований спосіб трансляції типів даних мови COBOL в типи даних сучасних мов програмування. Він базується на широкому використанні стандартних типів цільової мови, на відміну від більш простого підходу до реалізації семантики типів даних мови COBOL за допомогою системи додаткових класів.
Ключові слова: способи трансляції, типи даних, мова програмування COBOL, стандартні типи сучасних мов програмування.

Марченко А.И., Щербина Б.А. Способ трансляции типов данных языка COBOL в типы данных современных языков программирования. В статье предложен модифицированный способ трансляции типов языка COBOL в типы данных современных языков программирования. Он базируется на широком использовании стандартных типов целевого языка, в отличие от более простого подхода к реализации семантики типов данных языка COBOL с помощью системы дополнительных классов.
Ключевые слова: способы трансляции, типы данных, язык программирования COBOL, стандартные типы современных языков программирования.

Marchenko O.I., Shcherbyna B.O. Technique for COBOL types translation into types of modern programming languages. A modified technique for COBOL types translation into types of modern languages is proposed in this article. It is based on the wide usage of native types of the target language, rather than simplier implementing COBOL data types semantics using a set of additional classes.
Keywords: translation techniques, data types, COBOL programming language, native types of modern languages.

Вступ. Багато програм для управління технологічними та економічними процесами створені з використанням дещо застарілої в сучасних умовах мови COBOL. Для мови COBOL характерна структурна методологія програмування, яка призводить до складного процесу розширення функціональності програми. Найбільш поширеними на даний момент мовами програмування є об'єктно-орієнтовані мови, які забезпечують більш простий і швидкий процес модифікації і нарощення функціональності програмних систем. За допомогою мови COBOL нові програмні продукти вже не створюються, проте існує велика кількість програм, написаних на цій мові, працездатність яких необхідно постійно підтримувати, а в багатьох випадках також і розширювати.

Одним з підходів до розширення функціональності коду, написаного мовою COBOL є створення фактично нового програмного продукту з використанням однієї із сучасних мов програмування. Але такий підхід призводить до значних часових та фінансових витрат. Інший підхід полягає в розробці транслятора, що виконує автоматичне перетворення програм на мові COBOL в програми, що написані сучасними мовами високого рівня. Постановка задачі для таких мовних перетворень достатньо проста: необхідно перевести програму з однієї мови програмування на іншу, залишивши при цьому незмінним її зовнішню поведінку. Проблема такої трансляції залежить від наявності в цільовій мові програмування відповідних вбудованих мовних конструкцій, тобто при перекладі програм з однієї мови на іншу необхідно співставити набори вхідних та вихідних зразків кода, що і складає основну складність мовних перетворень.

Постановка наукової проблеми. Метою даної роботи є аналіз існуючих рішень з трансляції типів даних мови COBOL в типи даних сучасних мов програмування, а також розробка нового способу для такої трансляції.

Аналіз досліджень.

Типи даних мови COBOL

У мові COBOL всі структури даних оголошуються в спеціальному розділі програми, що називається DATA DIVISION або ж розділ даних. Дані можуть бути оголошені ієрархічно, кожне оголошення змінної має число-рівень, що вказує на належність змінної до іншої змінної. Оголошення змінної з більшим числом-рівнем вказує на її належність до змінної з меншим числом-рівнем. Така змінна, що містить в собі інші змінні, має назву RECORD, або "запис".

Саме оголошення змінної, окрім числа-рівня, містить ім'я змінної, її тип або формат, та необов'язкове початкове значення. Крім того, можуть бути вказані також спеціальні індикатори того, що змінна є масивом, або ж перевизначає іншу змінну. Зупинемось докладніше на форматі подання даних, що визначається конструкцією PICTURE.

Тип змінної в мові COBOL визначається описом значення, яке вона може містити, за допомогою спеціальних символів. Кожний символ відповідає за одну комірку пам'яті та позначає формат даних, що може в ній міститися. Найчастіше використовуються наступні позначення:

9 — комірка для цифри;

Z — комірка для цифри, що не відображається, якщо число починається нулем;

X — комірка для символа;

S — комірка для знака змінної;

V — комірка для символа коми в числі.

Наприклад, позначення 99999 означає змінну, що може містити до п'яти цифр, тобто така змінна є числом з п'яти цифр. Аналогічно формат XXXXXXX є поданням змінної, що складається з 7 символів, тобто рядком довжиною до 7 символів.

Важливим фактом в семантиці мови COBOL є те, що значення довільної змінної може бути присвоєне будь-якій іншій змінній, включаючи присвоєння окремого числа в масив. В такому випадку відбувається неявна конвертація даних з формату першої змінної в формат іншої змінної.

Одним з існуючих трансляторів з мови COBOL у мову Java є RES (An Open Cobol To Java Translator). Основною ідеєю трансляції типів даних у цьому трансляторі є створення класів, відповідних кожному типу даних, та організація взаємодії цих класів, наприклад, арифметичні операції реалізовані за допомогою певних методів. Ці згенеровані класи містять бінарне представлення даних, доступ до яких відбувається через методи цього класу. Такий підхід дозволяє повністю змоделювати поведінку типів даних мови COBOL, оскільки класи містять саме бінарне подання даних — подібним чином дані зберігаються і в мові COBOL.

Однак код, що згенерований таким способом, має два суттєвих недоліки. По-перше, такий код є досить громіздким через необхідність генерації всієї системи класів та методів для них. По-друге, змінні навіть найбільш примітивних типів, наприклад такі, що складаються з пари цифр, в такій програмі перетворяться у об'єкти класів із десятком методів для подання цих даних у різних форматах. Це призводить до суттєвого ускладнення потенціально досить простого коду, і як наслідок, більш складної подальшої роботи з ним.

На відміну від існуючих підходів, в роботі пропонується спосіб, який реалізує пряму трансляцію типів даних мови COBOL в стандартні типи даних цільових мов високого рівня. Під стандартними типами маються на увазі базові типи мов високого рівня, що використовуються для зберігання даних елементарних форматів: цілих і дробових чисел, символів і рядків.

У випадку мови COBOL подібні перетворення не можна виконати без додавання спеціальної логіки для деяких операцій, наприклад, округлення до певного знаку після коми, конвертації та приведення різних типів, тощо.

Виклад основного матеріалу й обґрунтування отриманих результатів дослідження.

Спосіб трансляції, що пропонується, складається з наступних кроків.

1. Програму, що написана мовою COBOL, транслюють в проміжне представлення, що напряму відображає поведінку та типи даних мови COBOL за допомогою спеціальної системи уніфікованих проміжних класів, таких як:

```
Interface Movable;
Class PicData<...> : Movable;
Class CobolArray<...> : Movable;
Class BinRep : Movable;
```

Тут інтерфейс Movable служить для відтворення семантики присвоєння значення довільної змінної в будь-яку іншу змінну. Параметризований клас PicData призначений для подання простої змінної. Параметром цього класу є "кобольне" подання змінної, тобто подання за допомогою

конструкції PICTURE. Параметризований клас CobolArray слугує для представлення масивів, де параметром класу є власне тип масиву. Клас BinRep (binary representation) слугує для внутрішнього подання даних у єдиному бінарному форматі, тобто саме у такому форматі, який використовується трансляторами мови COBOL.

Всі класи містять методи для конвертації в тип BinRep, а також конструктори, що приймають цей тип. Це необхідно для відтворення семантики присвоєння мови COBOL.

Таким чином, всі прості змінні транслюють в об'єкти класу PicData, а всі змінні-масиви — в об'єкти класу CobolArray. Всі записи ієрархічно транслюють в класи, а змінні, в такому випадку, є полями відповідних класів.

2. Типи змінних мови COBOL перетворюють в типи, що використовуються в заданій мові високого рівня. Відповідність перетворення типів на прикладі мови C# представлена в таблиці 1:

Таблиця 1. Таблиця відповідності типів мов COBOL та C#

PIC clause	Scope	Corresponding Type in C#
S9(n) $1 \le n \le 9$	- 999,999,999 ÷ 999,999,999	int
S9(n) $10 \le n \le 18$	- 999,999,999,999,999,999÷ 999,999,999,999,999,999	long
S9(m)V9(n) $1 \le m + n \le 7$	7 decimal digit precision	float
S9(m)V9(n) $8 \le m + n \le 15$	15 decimal digit precision	double
S9(m)V9(n) $16 \le m + n \le 28$	28 decimal digit precision	decimal
9(n) $1 \le n \le 9$	0 ÷ 999,999,999	unit
9(n) $10 \le n \le 18$	0 ÷ 999,999,999,999,999,999	ulong
9(m)V9(n) $1 \le m + n \le 7$	7 decimal digit precision	float
9(m)V9(n) $8 \le m + n \le 15$	15 decimal digit precision	double
9(m)V9(n) $16 \le m + n \le 28$	28 decimal digit precision	decimal

3. Операції над змінними мови COBOL перетворюють на операції, що визначені для типів заданої мови високого рівня. При цих перетвореннях повністю зберігається семантика роботи з даними мови COBOL. Конвертація типів виконується за допомогою класу BinRep, який має в собі інформацію, необхідну для трансформування даних відповідно до семантики мови COBOL.

4. За допомогою додаткового аналізу виконують трансформації для подальшого спрощення коду, якщо потрібно. Головним чином, ці трансформації спрямовані на заміну конструкцій з використанням класу BinRep на прості конструкції заданої мови високого рівня. Основними конструкціями для покращення є конвертація типів, а також операції з рядками.

Приклад

Для прикладу розберемо трансформацію фрагменту коду, що включає в себе семантичну конструкцію мови COBOL під назвою "модифікація за посиланням". Подібна конструкція передбачає зміну деяких розрядів змінної, не змінюючи при цьому інших. У сучасних мовах програмування подібні дії можна виконати над деякими структурами даних, але для цілочислених типів подібна операція не є визначеною. Відтак, задача трансляції подібної конструкції передбачає використання додаткових конструкцій для відтворення семантики початкової конструкції. Отож, фрагмент коду мовою COBOL:

```
01 date PIC 9(8).
01 year PIC X(4).
01 month PIC X(2).
01 day PIC X(2).

….
```

```
MOVE year TO date(1 : 4).
MOVE month TO date (5 : 2).
MOVE day TO date (7 : 2).
```

В наведеному прикладі у перші чотири розряди восьмирозрядної змінної date записується значення змінної year, в п'ятий та шостий розряди — значення month і в сьомий та восьмий — значення змінної day. Використовуючи наведену схему трансляції згенерований код буде поетапно виглядати наступним чином:

```
{
        BinRep __refmod__ = null;
        __mt_refmod__ =
        CobolUtils.toBinRep("9(08)", date).subrepmod(PicData<S9>(BinRep(-1)),
PicData<9>(BinRep(1)).add(PicData<9>(BinRep(4))).sub(PicData<9>(BinRep(1))), CobolUtils.toBinRep("X(04)", year));
        date =__mt_refmod__.subrep(PicData<9>(BinRep(0)), PicData<9>(BinRep(0)) +
PicData<9>(BinRep(8))).asuint32_t("99999999");
}
{
        BinRep __refmod__ = null;
        __refmod__ =
        CobolUtils.toBinRep("9(08)", date).subrepmod(PicData<9>(BinRep(5)).sub(PicData<9>(BinRep(1))),
PicData<9>(BinRep(5)).add(PicData<9>(BinRep(2))).sub(PicData<9>(BinRep(1))), CobolUtils.toBinRep("X(02)", month));
        date = __mt_refmod__.subrep(PicData<9>(BinRep(0)),
PicData<9>(BinRep(0)).add(PicData<9>(BinRep(8)))).asuint32_t("99999999");
}
{
        BinRep __refmod__ = null;
        __refmod__ =
        CobolUtils.toBinRep("9(08)", date).subrepmod(PicData<9>(BinRep(7)).sub(PicData<9>(BinRep(1))),
PicData<9>(BinRep(7)).add(PicData<9>(BinRep(2))).sub(PicData<9>(BinRep(1))), CobolUtils.toBinRep("X(02)", day));
        date =__mt_refmod__.subrep(PicData<9>(BinRep(0)),
PicData<9>(BinRep(0))).add(PicData<9>(BinRep(8))).asuint32_t("99999999");
}
```

На другому та третьому етапі відбувається заміна типів, що відповідають простим об'єктам на типи мови високого рівня. В результаті цього етапу перетворення фрагмент буде виглядаті наступним чином:

```
{
        BinRep __refmod__ = null;
        __mt_refmod__ =
        CobolUtils.toBinRep("9(08)", date).subrepmod(1 -1,1 +4 -1, CobolUtils.toBinRep("X(04)", year));
        date =__mt_refmod__.subrep(0, 0 + 8).asuint32_t("99999999");
}
{
        BinRep __refmod__ = null;
        __refmod__ =
        CobolUtils.toBinRep("9(08)", date).subrepmod(5 -1,5 +2 -1, CobolUtils.toBinRep("X(02)", month));
        date = __mt_refmod__.subrep(0, 0 + 8).asuint32_t("99999999");
}
{
        BinRep __refmod__ = null;
        __refmod__ =
        CobolUtils.toBinRep("9(08)", date).subrepmod(7 -1,7 +2 – 1, CobolUtils.toBinRep("X(02)", day));
        date =__mt_refmod__.subrep(0, 0 + 8).asuint32_t("99999999");
}
```

За семантикою мови відбувається присвоєння у відповідні розряди через внутрішнє подання даних, що в наведеному прикладі імітується використанням класу BinRep. Таку конструкцію можна істотно спростити, розпізнавши, що присвоєння відбувається у одну і ту ж змінну, а її розряди оновлюються послідовно. Таким чином, можлива заміна на лише одну операцію, що передбачає конкатенацію рядків і їх конвертацію в число. В результаті присвоєння виду:

```
date = CobolUtils.toBinRep(year + month + day).asuint32_t("99999999");
```

У такому варіанті змінні year, month і day транслюються в стороковий тип (string), потім відбувається їх конкатенація і перетворення у цілочислений тип змінної date.

Висновки

Код, що генерується запропонованим способом має більш чітку та інтуїтивно зрозумілу структуру і семантику, а також є суттєво меншим за розміром. Конструкції, що додаються в п.1, в загальному випадку мають бути замінені на конструкції мови високого рівня. Враховуючи широкий спектр можливих комбінацій таких конструкцій, а також істотний семантичний розрив мови COBOL із сучасною методологією програмування, основною задачею для подальшого вдосконалення способу є розробка ефективних трансформацій для їх перетворення на етапі описаному у п.4.

1. RES – An Open Cobol To Java Translator / Sourceforge [Електронний ресурс]. Режим доступу: http://sourceforge.net/projects/opencobol2java/
2. Enterprise COBOL for z/OS / IBM [Електронний ресурс]. Режим доступу: http://www-03.ibm.com/software/products/en/entecoboforzos
3. Свердлов С.З. Языки программирования и методы трансляции: Учебное пособие. - СПб.: Питер, 2007. - 638 с.
4. Описание синтаксиса языков программирования [Электронный ресурс]. Режим доступа: http://wiki.mvtom.ru/index.php
5. Терехов А., Верхув К. Проблемы языковых преобразований [Электронный ресурс]. Режим доступа: http://www.osp.ru/os/2001/05-06/180189/
6. Widmer R. COBOL Migration Planning, Edge Information Group, 1998.
7. Y. Chae and S. Rogers, Successful COBOL Upgrades: Highlights and Programming Techniques, John Wiley and Sons, New York, 1999
8. Молдованова О.В. Языки программирования и методы трансляции: Учебное пособие. – Новосибирск, 2012. – 134 с.
9. Ахо А.В. Компиляторы: принципы, технологии и инструментарий / [Ахо А.В., Лам М.С., Сети Р., Ульман Дж.Д.] 2-е изд. : Пер. с англ. – М.: Изд. дом Вильямс, 2008. –1184 с.
10. Fast Lexical Analyzer. [Электронный ресурс]. Режим доступа: http://flex.sourceforge.net/

УДК 004
Melnuk V.M., Melnuk K.V., Khrystynets N.A
Lutsk National Technical University

KEY PERFORMANCE INDICATORS BASED SOFTWARE ECOSYSTEM (SECO) RESEARCH USING PUBLICATIONS SYSTEMATIC MAPPING

Melnuk K.V., Melnuk V.M., Khrystynets N.A. Key performance indicators based software ecosystem research using publications systematic mapping. To create value with a software ecosystem, a platform owner has to ensure that the SECO is well and sustainable. Key Performance Indicators (KPI) are used to assess whether and how well such objectives are met and what the platform owner can do to improve. This paper gives research overview on KPI-based SECO assessment using research publications systematic mapping. The study identified 34 publications for which KPI research and KPI practice were extracted and mapped. It describes the strengths and gaps of the research published later and what KPI are measured, analyzed, and used for decision-making from the researcher's point of view. The maps that capture state-of-knowledge can be used to plan further research. For practitioners, the generated map points to studies that describe how to use KPI for SECO managing.
Keywords: Software ecosystem, digital ecosystem, performance indicator, KPI, success factor, systematic mapping.

Мельник К.В., Мельник В.М., Христинец Н.А. Ключевые индикаторы функционирования входящие в исследование экосистемы программного обеспечения, используя систематическое картографическое отображение. Чтобы создать значение с экосистемой программного обеспечения, владельцу платформы придется гарантировать, что SECO хороший и работоспособный. Ключевые показатели деятельности (KPI) использованы, чтобы оценить, насколько хорошо выбраны цели и что владелец платформы может сделать для улучшения. Эта статья дает исследовательский краткий обзор оценке SECO на базе KPI, пользуясь систематической картографией исследовательских публикаций. Изучение идентифицировало 34 публикации, для которых были извлечены и изображены исследование и практика KPI. Это описывает силы и пробелы исследования, изданного позже и что KPI, есть взвешенный, проанализировавший, и использованный для принятия решения с точки зрения исследователя. Изображения содержат взятый уровень знаний и может быть использовано, чтобы планировать дальнейшее исследование. Для специалистов-практиков производимая карта указывает на изучение, которое описывает, как пользоваться KPI для SECO.
Ключевые слова: экосистема программного обеспечения, цифровая экосистема, показатель деятельности, KPI, фактор успеха, систематическая картография.

Мельник К.В., Мельник В.М., Христинець Н.А. Ключові індикатори функціонування що входять в основу дослідження екосистеми програмного забезпечення, користуючись систематичною картографією публікацій. Щоб створити значення з екосистемою програмного забезпечення, власникові платформи доведеться гарантувати, що SECO хороша і роботоздатна. Ключові показники діяльності (KPI) використані, для оцінки чи добре підібрані є цілі і що власник платформи може зробити для поліпшення. Ця стаття дає короткий дослідницький огляд на основі оцінки KPI SECO, користуючись систематичною картографією дослідницьких публікацій. Вивчення ідентифікувало 34 публікації, для яких були витягнуті і зображені дослідження і практика KPI. Це описує сили і недоліки дослідження, виданого пізніше і що KPI зрівноважений, аналізуючий, і використаний для прийняття рішення з точки зору дослідника. Зображення містять виявлений рівень знань і можуть бути використані для планування подальших досліджень. Для практикуючих фахівців карта відображення вказує на вивчення, яке описує, як користуватися KPI для SECO.
Ключові слова: екосистема програмного забезпечення, цифрова екосистема, індикатор діяльності KPI, фактор успіху, систематична картографія.

Introduction

A software ecosystem (SECO) is about actors set interaction functioning as a unit and interacting with a shared market for software and services together with the relationship among them [1]. Here is reviewed any ecosystem that basing on or enabled by software, including pure software, software-intensive systems, mobile applications, cloud, telecommunications, and digital software ecosystems. The inclusion of telecommunications, for example, can only be realized with appropriate ICT infrastructure. Companies adopt SECO strategy to enlarge their organizational boundaries, to share their platforms and resources with third parties and to define new business models [2, 3]. A SECO is often supported by a technological platform or market that enables the SECO actors in exchanging information, resources, and artifacts. Ownership of such a platform gives strategic advantages over the other SECO actors. It allows ever-increasing customer demands satisfaction with limited own resources. It also KPIs for Software Ecosystems: A Systematic Mapping Study 195 allows improving one's own knowledge about the marketplace that is necessary for innovation, evolution of a product or service offering, and revenue opportunities identification [4, 5].

SECO platform ownership also brings responsibilities which include the definition of SECO performance objectives and SECO management to achieve these objectives. It also is expected to be healthy [6] and sustainable [7]. It is healthy when it is productive for surrounding actors, robust, and niche-creating [8]. It is sustainable when it maintains its structure and functioning in a resilient manner

[6]. Health and sustainability are closely linked performance objectives that are often found in complex systems [9, 10]. Managing involves definition of how actors, software, and business models play together to achieve the SECO objectives in business, technical, and social dimensional perspectives [11, 12]. The platform owner uses performance indicators for benchmarking and monitoring the resulting ecosystem behavior. Key performance indicators (KPI) are those among many possible indicators that are important, easily measurable quantitatively or with a qualitative phenomenon approximation [13]. The KPI serve as early warnings about potentially missed SECO objectives [14] and used to detect patterns that are useful for predicting health and sustainability [15]. Any deviation from success baselines are recorded and acted upon to ensure about main ecosystem's objectives are met. The presented study gives a literature overview of KPI for software ecosystems. A systematic mapping methodology was surveyed to identify and classify publications based on the reported research and on KPI use. The used for classifying research dimensions were the studied type of ecosystem and the result type was delivered by the research. The dimensions used for classifying KPI use were investigated KPI types, the SECO objectives were used these KPI for. The knowledge gap for collecting evidences about KPI studies motivated systematically evaluate distribution of studies and provide guidance for future improvement. For practitioners, the generated map describes how to use KPI in the SECO management. It enables the platform owner in indicators understanding that are important to assess given SECO objectives. For researchers, the generated map describes research state and helps finding research gaps for definition understanding and SECO KPI use.

Research Methodology

The goal of this study is to provide research overview performed to investigate the use of KPI for managing software ecosystems. The systematic mapping approach [16] allows to map the frequencies of publications over categories to see the current state of research. It also exposes patterns or trends of what research kind is done or respectively has been ignored so far. The research results mapping in addition to the research type reveals researchers' current understanding of KPI-related practice.

To provide an overview on publications relevant to KPI use for SECO, two sets of research questions are defined in Table 1. With the first set of questions we mapped foci and gaps for SECO KPI research. With the second set we mapped the practice state that was reported by the research.

Research Questions.

SECO KPI Research Rationale. RQ1: What kinds of ecosystems were studied? The answer to this question shows the SECO KPI research intensity across domains and types of ecosystem application. Due to a focus on just a few types of application domains and ecosystems, skewedness indicates gaps where additional research is need. RQ2: What types of research were performed? The answer to this question shows the maturity of SECO KPI research. The more disproportioned conceptual solutions and empirical validation research are, the more there is a need for research to compensate.

Ecosystem KPI Practice Rationale. RQ3: What objectives were KPI used for? The answer to this question shows the SECO KPI purposes. It allows understanding when a SECO is considered to be successful and not. The answer to RQ4 correlation allows understanding how the SECO objectives satisfaction is measured. RQ4: What ecosystem entities and attributes did the KPI correspond to? The answer to this question gives a relevant KPI overview that are used to assess SECO objectives achievement. The KPI show how SECO objectives are operationalized and quantified. Skewedness, focusing on just one or a few KPI, may indicate the degree of universality that KPI have for SECO management.

Systematic Mapping Approach

To answer RQ1, RQ3, we followed the systematic mapping guidelines proposed in [16]. We: conducted database search with a search string matching to our research scope; performed screening to select the relevant papers; built a classification scheme based on keywording the paper titles, abstracts, and keywords; and used this classification scheme to map the papers. To answer RQ2, we modified the mapping process by using the pre-existing classification schemes already used in [16, 17]. For RQ4, we built the classification scheme by extracting keywords from the main body of the papers and aligning the emerging scheme with the relevant software industry standard. The research steps are explained below.

I Database Search. The study defined the following search strategy.

Search String. To get an unbiased overview of KPI use in SECO, the search string was created with keywords that capture population only. The first aspect used to define the population was the ecosystems that can be found in a software context: software, digital, mobile, service, cloud, telecommunication and

ICT ecosystems. We also included papers that focused on software supply by adding software supply to the search string. The second aspect used to define the population was the KPI application or use. It was used the term indicators, metrics, measurements, success factors, key characteristics and quality attributes as synonyms for KPI. To avoid bias about RQ3, we did neither constrain for what purpose information was gathered and used. To build a broad overview of the research area and avoid bias, no keywords were defined in relation to intervention (e.g. monitoring), outcomes (e.g. improvements to SECO), or study designs (e.g. case studies). The search string was built by concatenating the two population aspects with the *AND* operator. The search string was formulated as follows:

Software OR *(software-intensive)* OR *digital* OR *mobile* OR *service* OR *cloud* OR *communic** OR *telecom** OR *ICT PRE/0 (ecosystem** OR *"supply network*")* AND *(measure** OR *kpi** OR *metric** OR *analytic** OR *indicator** OR *"success factor*"* OR *"quality attribute*"* OR *"key characteristic*"*.

Search Strategy. The papers were identified using the important research databases in software engineering and computer science including Scopus, Inspec, and Compendex, which support IEEEXplore and ACM Digital Library as well. The search string was applied to title, author's keywords and papers abstract. The search did not restrict the publication date.

Validation. It was validated the identified papers set by checking it against the papers used in the SECO literature reviews performed by [2, 5]. Each paper used by these studies that was relevant for our study had been found by following the above-outlined database search.

II Screening of Papers. The inputs for this step were the set of papers identified with step (I). The first and second authors screened these papers independently. It was screened these papers to exclude studies not related to the KPI use for any ecosystem-related purpose and to ensure broad-enough coverage of the population. Next a complete set of inclusion and exclusion criteria is described.

Inclusion. It was included peer-reviewed journal, conference, or workshop papers that were accessible with full text. The included papers describe the use of KPI in an ecosystem context or the effects of such KPI on ecosystem properties. Due to the importance of networking infrastructure and digital information exchange for a well-functioning software ecosystem there was included telecommunication and information technology papers in addition to pure SECO papers.

Exclusion. There was excluded papers that focused on the KPI use for managing an ecosystem member only. For example, papers about the indicators use for managing a single company that participates in the ecosystem or a product or company process were excluded because of their too narrow focus. It was also excluded papers that focused on other ecosystems rather than a software ecosystem. For example, papers focused on biology, environmental, climate, and chemical aspects were excluded. When the software ecosystem definition did not fulfill in the papers, they were excluded. As an example, the paper that considered Bugzilla and email system as software ecosystems was excluded, since such systems do not address the shared market concept of SECO definition. Papers that studying qualitative indicators using qualitative approaches such as a structured interview were too excluded. Also, it was excluded papers that focused on ecosystem design in place of ecosystem management. For example, papers about the design of interoperability protocols, products, services offered to ecosystem were excluded. To avoid inclusion of papers that only speculated about KPI use or effects, it was excluded papers that did not report any empirically-grounded proof-of-concept.

III Building the Classification Scheme. To answer the research questions RQ1, RQ3 and RQ4 it was applied keywording [16] as a technique to build the classification scheme in a bottom-up manner. Extracted Keywords were grouped under higher categories to make categories more informative and to reduce number of similar categories. The ecosystem classification scheme was built by extracting the types and application domains of the studied ecosystems. It was built the classification scheme for KPI practice by extracting KPI assessment objectives, entities and attributes used for measuring KPI. The keywords were extracted from the papers' titles, keywords, and abstracts. When the abstract quality was too poor, to identify the keywords was used the paper main body. Similarly, as most of the papers did not included sufficient information about entities and attributes measured with KPI inside the abstract, the main papers body was used for keyword identification. The keywords obtained from extraction were then combined and clustered to build the categories used for mapping the papers. The measurement attributes clustering was aligned with the categories described in ISO/IEC FDIS 25010 as far as applicable. To answer RQ2, it was used a pre-defined classification scheme [17] that was used by earlier systematic mapping studies [16]. It classifies research types into validation research, evaluation research, solution proposals, philosophical papers, opinion papers, and experience papers.

IV Systematic Mapping of the Papers. When the classification scheme was ready, the selected papers were sorted into the classification scheme. Then classifications calculated the frequencies of publications for each category. To answer RQ1 and RQ2 was reported the frequencies of the selected papers for the categories in the dimensions of ecosystems types and application domains, respectively in the dimensions of research type and research contributes type. We used x-y scatterplots with bubbles in category intersections to visualize the kinds of studied ecosystems. The size of a bubble is depicted proportional to the number of papers that are in the pair of categories corresponding to the bubble coordinates. The visualized frequencies make possible to see which categories have been emphasized in past research and which of them received little or no attention. To answer RQ3, first was described the categories identified when building the classification scheme and how these categories were expressed in the selected papers. This description resulted in a dictionary for interpreting the scatterplots used for describing how SECO KPI are used for these objectives. Again, x-y scatterplots were used for showing the frequency of categories pairs. These pairs allowed to describe the attributes measured for each type of ecosystem entity, the measurements used in relation to the SECO objectives, and how KPI are obtained for various kinds of entities found in SECO.

Threats to Validity

The threats to validity are analyzed for construct taxonomies, reliability, internal and external validity. *Construct validity* reflects whether the papers included in the study reflect the SECO KPI phenomenon that was intended to be researched. The search string was constructed in an inclusive manner so that it captured the wide variety of software-related ecosystems and the many different names given to key performance indicators. The common databases, used for software and management-related literature research, were used to find papers. Only after this inclusive process, manual screening was performed to exclude papers not related to the research objectives. The list of included papers was then validated against two systematic studies on software ecosystem [2, 5] and found that the review covers all relevant papers.

Reliability validity refers to the study repeatability for other researchers. The study applied a defined search string, used deterministic databases, and followed a step-by-step procedure that can be easily replicated. The stated inclusion and exclusion criteria were systematically applied. Reliability of the classification was achieved by seeking consensus among multiple researchers.

Internal validity treats refers to problems in the data analysis. These threats are small, since only descriptive statistics were used.

External validity concerns the ability to generalize from this study. Generalization is not an aim of a systematic mapping study as only one research state is analyzed and the relevant research body completely covered. In particular, the study results about the SECO KPI use reflects the practices studied in SECO KPI research and not SECO KPI practice performed in general.

Ecosystem KPI Research Results

The database search resulted 262 papers in total, including 46 duplicates. After screening and exclusion 34 papers remained and were included in the study. Selected papers were published from 2004 onwards. It will be given an overview of the research described in the selected papers and app. A lists the selected papers.

Kinds of Ecosystems. To answer RQ1, Figure 1 gives an overview over the ecosystems that our study found for KPI research. The number embedded in a bubble indicates how many papers were devoted to a given combination of ecosystem type and application domain (multiple classifications possible). Empty cells indicate that no corresponding study was found. The number on the category label indicates the total number of papers in that category. Most of the papers used the term software ecosystem to characterize the studied ecosystems. Special kinds of ecosystems were cloud, service, mobile apps, and open source software ecosystems. Less frequent were digital ecosystems with 44% of the papers. They refer to the use of IT to enable collaboration and knowledge exchange [16]. The papers addressed a variety of application domains. Most common were telecommunications, business management and software development. None of the remaining application domains was addressed by more than one or two papers. Thus, research is rather scattered, and the specifics of the various application domains understood only little.

Types of Research. To answer RQ2, Fig. 1 presents a map of the research kinds performed on KPI in software-related ecosystems. Papers with multiple research types and contributions were classified for each research type combination and contribution they presented.

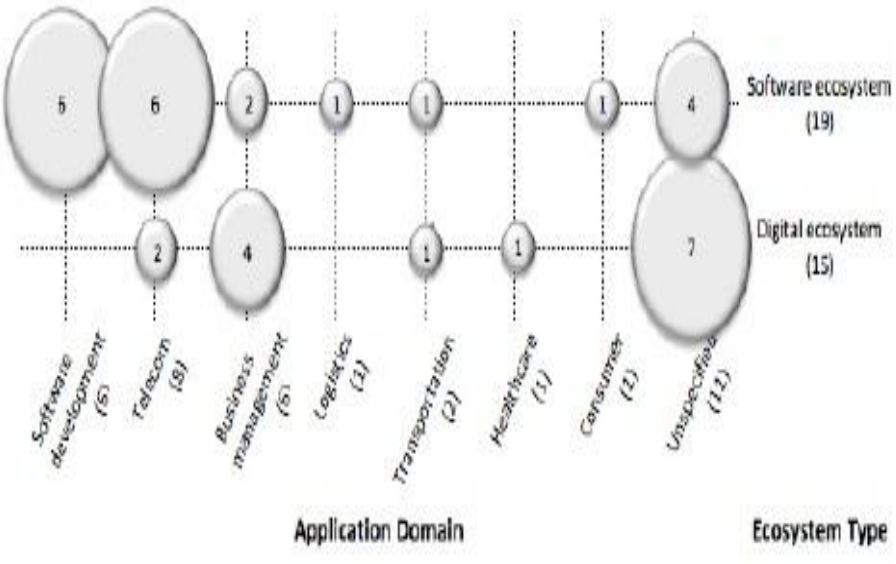


Fig. 1. Map of research on SECO KPI and type of contributions

Experience report papers describe experiences in working with SECO KPI and unsolved problems.
Opinion papers discuss opinions of the papers' authors.
Conceptual proposal papers sketch new conceptual perspectives related to SECO KPI. This category renamed *philosophical papers* category (described in III) to fit the SECO KPI study.
Solution proposal papers propose new methods or improve existing techniques using a small example or a good argumentation.
Validation papers investigate novel solutions that had not been implemented in practice (e.g. experiment, lab working).
Evaluation papers report on empirical or formal studies performed to implement a solution or evaluate the implementation.
Metric papers describe KPI for SECO.
Model papers describe relationships between KPI.
Method papers describe approaches for working with SECO KPI.
Tool papers describe support for work with SECO KPI.

Most research were found in the validation and evaluation categories. Research contributed with metrics, models, or methods. For example, R17 proposes a model that explains how health can be measured with relevant indicators (conceptual proposal, model) and validates that model with a questionnaire (validation, model). R14 proposes a method for assessing services based on Service indicators Quality (solution, method). R19 evaluates factors that affect successful selling in e-markets (metric, evaluation). No paper was with the experience report or an opinion paper and no paper were contributed with any tool.

Researched KPI Practice Results

In this study the papers included described use of KPI by a platform owner for achieving objectives with the ecosystem that was enabled by the ecosystem platform. It is given an overview of these objectives and used KPIs.

Ecosystem Objectives Supported by KPI. KPI were used to enable or achieve a variety of objectives. Platform owners aimed, at improving business, interconnectedness between actors and quality of ecosystem, product, or services performed within the ecosystem, at ecosystem growing and at enabling ecosystem sustainability (answer RQ3):

Business improvement. Research has been made on how to improve business at the ecosystem level. The studied business improvements concerned the perspectives of ecosystem activity and commercial success. Ecosystem activity related to the activity level of participating actors, encouragement to participate in the ecosystem, and the transaction volume. Commercial success related to

sales success, innovativeness and competitiveness of the participating actors, and the network cost that enables the ecosystem. The activity and commercial perspectives were mixed in the papers, thus could not be separated in the literature analysis.

Interconnectedness improvement. Research has been performed on how to improve interaction in an ecosystem, for example to reduce cost, improve predictability of services that are provided in the ecosystem, and manage trust. Interaction improvement has been studied between individual actors and between whole networks contained in the ecosystem. The research differed in lifecycle stage terms of an interaction and covered supplier availability, discovery, ranking and selection, the resulting connectivity, interaction evaluation, and the interaction actors' impact that participated in it. Interaction improvement was considered essential for generating business activity and ecosystem sustainability.

Growth and stability. Research has been made on how to manage ecosystem growth and stability. They were seen as two factors that have to be managed jointly. During growth flexibility and controllability has to be maintained. During stability, a continuous co-revolution must take place. Growth and stability again are not ends in themselves, but thus contribute to ecosystem sustainability and survival.

Quality improvement. Research has been performed on how to manage quality of ecosystems. In particular, performance, usability, security, data reliability, extendibility, transparence, trustworthiness, and quality-in-use were investigated here. Quality management was sometimes presented as an ends in itself, for example by allowing comparison among multiple ecosystems, enabling diagnosis, improving decision-making, and achieving services long-term usage. At the same time, however, quality management was considered to be a means to encourage adoption and growth, improve business performance, and achieve sustainability.

Enable sustainability. Research has been made on how to sustain an ecosystem. Two angles were taken: self-organization and resource consumption. Self-organization was approached through continuous ecosystem rejuvenation. Resource consumption was studied in relation of electrical energy. Throughout all papers found in this category, sustainability was considered to be desirable ends for software ecosystems.

KPI Measured Entities. Included papers describe measurements applied to the ecosystem as to the parts the ecosystem consists of: actor, artifact, service, relationship, transaction and network.

Actors were measured and characterized as follows. They were human or artificial. Examples of human or legal actors were sellers and developers that provide products to buyers or organizations and companies groups. Examples of artificial actors were nodes in a telecommunication network. An actor engages in transactions in ecosystem and builds relationships to other actors or artifacts. The transactions engages the seller in generate profit and revenue for the cost the seller is keen to take. Effective actors have knowledge about other actors or network, good interestingness and reputation for other actors. Actors are also considered to be sources and sinks of data and have differing ranges for data transmission. Performance of individuals and groups in terms of fulfilled tasks and decisions as well as firms and organizations performance in measured terms of profits.

Artifacts, such as software, codes, plugins, books, music, or data were measured and characterized as follows. Artifacts had a location in the ecosystem. They evolve, may have reputation and popularity, and exposed their consumers to vulnerability.

Services were measured and characterized as consuming energy and other resources. Services have quality attributes: service quality, security, compliance and reputation. Metadata and service level agreements are used to specify the services. The services are not fixed but evolve: services emerge, change, and get extinct. A special service was provided by the platform that laid the fundament for the ecosystem. It was characterized in attributes terms like stability, documentation, portability, and openness.

Relationships were measured and characterized as follows. Actors enter relationships with other actors, artifacts, or services. A relationship connects two or more such entities. Relationship examples were business connections and telecommunication communication links. A relationship may be transparent and express a trust value of the connected entities. A relationship is the basis for transactions, thus is used for advertising and building alliances. The transaction, however, is constrained by relationship cost and quality.

Transactions were measured and characterized as follows. Examples of transactions are services sales to customers, server requests, and code files commits made by developers. They are initiated with an

offer that is measured in attributes terms like price and quantity. Transactions also have a price and quantity. Other attributes include time to negotiate the transaction, time to complete, energy consumption, transmission rate, and buyer satisfaction.

Networks were considered as sets of entities and relationships that were part of a whole ecosystem. Examples were local or application-specific networks. Networks were vulnerable to security threats such as data availability, integrity, authentication and authorization. They differed in the node density, collaboration degree, provisioning cost, and hit rate for artifacts.

Ecosystem. Full ecosystems have quality attributes like size, performance, security and energy consumption that can also characterize networks contained in an ecosystem. In addition, ecosystems exhibited lifelines, diversity, stability, transparency, healthiness and sustainability. This section and next one collaboratively provide an answer for RQ4. The map in the left part of Figure 3 shows the entities that were studied in relation to the ecosystem objectives. Most research studied the overall ecosystem measurement to enable quality or business improvement. For example, R17 describes how performance of the ecosystem affected user satisfaction, and R13 shows how analytics applied to ecosystem can be used to improve business. Considerable research was also devoted to ecosystem interconnectedness improving, where products attributes and services played an important role including for platform measurements to grow the ecosystem and improve its quality. For example, R6 described how a service similarity measurement was used to improve ecosystem connectivity. R2 described how growth, diversity, and entropy measurements of a SOA platform were used to increase growth. R4 described how communication quality measurements were used to improve the telecommunication ecosystem quality. The map also shows areas where no research was published. For example, no one research studied the network measurements role for objectives other than sustainability and quality improvement.

KPI Measurement Attributes

To make the state and evolution of the ecosystem and its elements visible, a broad variety of attributes were measured. The following attributes categories emerged when clustering the attributes described in the included papers. Fig. 4 shows how quality attributes classes were merged toward new categories. The *size* category includes attributes to measure size and growth. *Diversity* includes attributes to measure heterogeneity and openness for such heterogeneity. *Financial* includes attributes to measure economic aspects such as investment, cost, and price. *Satisfaction* includes attributes to measure it and the related concepts of suitability, interestingness, learnability, usability, accessibility, acceptability, trust, and reputation. *Performance* includes attributes to measure it, including resource utilization, efficiency, accuracy and effectiveness. *Freedom from risk* includes attributes to measure the ability to avoid or mitigate risks and includes the related security concerns, reliability, maturity, availability, and other related guarantees. *Compatibility* includes attributes to measure the degree to which an entity can perform well in a given context, interoperate or exchange information with other entities and be ported from one context to another. *Maintainability* includes attributes to measure flexibility, respectively the ability to be changed. The right part of Fig. 3 gives an overview of the attributes referred to KPI. Most research studied satisfaction measurements typically to improve business or interconnectedness. Such research example is R13 that describes the seller reputation use to improve business. To support quality improvement all measurement attributes related to quality were included in at least one research paper, except for maintainability and size. Similarly, size measurements did not play any role other than for growth and stability. The left part of Fig. 5 shows how the ecosystem elements were measured. Satisfaction was a common attribute that was measured for any entity except for rules. This shows that a same attribute can be measured or analyzed for different ecosystem entities. It is also revealed that similar measurement attributes might be collaborating to measure different ecosystem elements. As an example correlation, commitment, clarity and importance (CCCI) measurable attributes were used to measure trust as well as reliability.

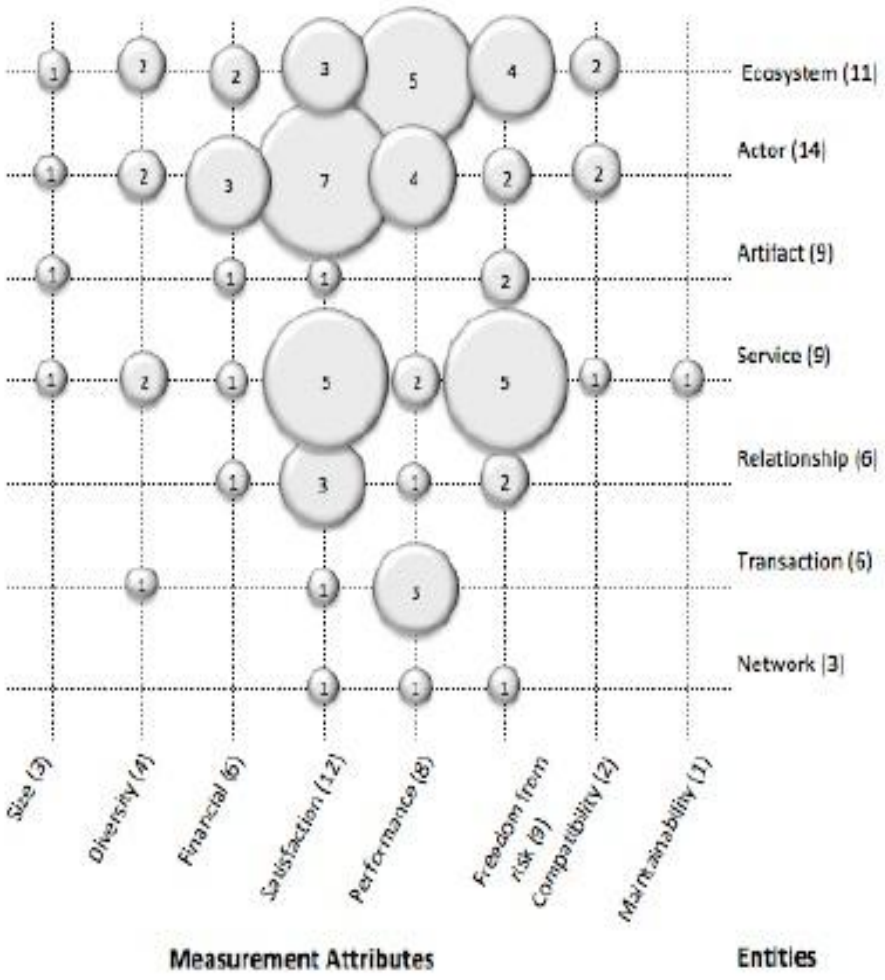
Diversity	*Satisfaction*	*Performance*	*Financial*	*Size*	*Freedom from risk*	*Compatibility*	*Maintainability*
Heterogeneity Openness	Learnability Usability Accessibility Acceptability Trust Reputation	Satisfaction Suitability Interestingness Performance Resource utilization Efficiency Accuracy Effectiveness	Investment Cost Price	Size Growth	Risk mitigation Security Reliability Maturity Availability Guarantees	Interoperability Exchangeability	Flexibility Changeability

Fig. 2. Measurement attributes merging classifications

The overall ecosystem was the most comprehensively measured or analyzed entity with a special focus on satisfaction, freedom from risks and performance. Some examples of such satisfaction measurements are provided by R13 that measured ecosystem usage and acceptability. The platform tailed with the second-largest variety of measurements. R2, for example, measured entropy and diversity to characterize platform complexity. Only narrow sets of measurement attributes were applied to the business partner, interactions, and business.

Discussion

The study provides a KPI relevant papers classification in understanding researches, relationship with the practice and research outcomes assessment. This classification contributes to taxonomy, which can help for closer examination of the ecosystem or platform owner objectives, making them more recognizable in designing KPI. New KPI can be extracted for an ecosystem using this taxonomy and existing KPIs can be extended or restructured applying the generic taxonomy structure. The literature map indicates that KPI for software-based ecosystems is a thin area with work at all maturity levels. Journal, conference and workshop papers exist. However, the number of publications is not sufficient and many application domains for ecosystems addressed with just one or two papers. Although KPI formulation might be domain dependent and similarity of objectives is not the only factor to select a KPI, however, due to insufficient study it is difficult to state whether domain characteristics, for example healthcare regulation, affects the ecosystem KPI that targets that domain. The included research on ecosystem KPI mostly addresses ecosystem measurements or satisfaction measurements, performance and freedom from risks. Measurements other than satisfaction that are applied on elements contained in the ecosystem are comparatively little researched. A KPI broader understanding would increase a platform owner's flexibility in measuring, analyzing, and using KPI for decision-support. The understanding of a greater KPI variety would also contribute to increased status transparency, evolution, and other ecosystem aspects.



Conclusion

Presented study gives literature overview for KPI use with software-based ecosystems. A systematic mapping methodology was followed and applied to 34 included studies published from 2004 onwards. To respond to RQ1 and RQ2, research was broad but thin. Two major kinds of ecosystems were researched: software ecosystems and digital ecosystems. Many application domains were addressed, but most of them with one or two papers only. The published research was mature with journal, conference, and workshop papers that covered metrics, models, and methods. In response to RQ3 and RQ4, KPI research was skewed. Most research studied ecosystem KPI for improving the interconnectedness between individual actors and subsystems of the ecosystem. Overall, most KPI were about satisfaction, performance and freedom from risks measures. The mapping study results indicate that more research is needed to better KPI understanding for software-based ecosystems. In particular, a deeper understanding of how the application domain affects the ecosystem's KPI is needed. Also, an important research opportunity is the identification, analysis, and evaluation of KPI. Such research could make the work with KPI more flexible, because a greater KPI variety would be known and available for the practitioner to use.

1. Andreas Meiszner Open Education Ecosystems, learning analytics and supportive software system framework (67 SlideShares), Founder & Managing Partner at SCIO – Sociedade do Conhecimento, Inovação, e Organizaçao LDA, Published on May 30, 2012
2. Jansen, S., Finkelstein, A., Brinkkemper, S.: A sense of community: A research agenda for software ecosystems. In: International Conference on Software Engineering. ICSE-Companion 2009, Vancouver, British Columbia, Canada (2009)
3. Manikas, K., Hansen, K.M.: Software ecosystems–a systematic literature review. Journal of Systems and Software 86, 71–80 (2012)
4. Weiblen, T., Giessmann, A., Bonakdar, A., Eisert, U.: Leveraging the software ecosystem: Towards a business model framework for marketplaces. In: 3rd International Conference on Data Communication Networking, DCNET 2012, 7th International Conference on e-Business, ICE-B 2012 and 3rd International Conference .

5. Dhungana, Deepak; Groher, Iris; Schludermann, Elisabeth; Biffl, Stefan. Software ecosystems vs. natural ecosystems: learning from the ingenious mind of nature., 2010 URI:http://hdl.handle.net/10344/2359, Publisher:Association for Computing Machinery,Publication type:info:eu-repo/semantics/conferenceObject.
6. A Survey of Associate Models used within LargeSoftware Ecosystems, Joey van Angeren, Jaap Kabbedijk, Slinger Jansen, and Karl Michael Popp,*Department of Information and Computing Sciences, Utrecht UniversityPrincetonplein 5, 3508 TB Utrecht, the Netherlands*.vanangeren,j.kabbedijk,s.jansen}@cs.uu.nl *SAP AG, Corporate DevelopmentDietmar-Hop-Allee 16, 69190 Walldorf, Germany,* karl.michael.popp@sap.com
7. Waters, Sandie H.; Gibbons, Andrew S. (2004). Design Languages, Notation Systems, and Instructional Technology: A Case Study Educational Technology Research and Development, 52, 2.
8. Watkins, Debbie; Kritsonis, William Allan (2008). Aristotle, Philosophy, and the "Ways of Knowing Through the Realms of Meaning": A National Study on Integrating a Postmodernist Approach to Education and Student Achievement [Online Submission]
9. M. Lehman. Software Evolution – Background, Theory, Practice. Integrated Design and Process Technology. Society for Design and Process Science. – 2003 – 11p.
10. Aid to recovery: the economic impact of IT, software, and the Microsoft ecosystem on the global economy. – IDC White Paper, – 2009, 9p.
11. Cynthia Keeshan. The Software Ecosystem. - http://www.microsoft.com/ canada/media/ecosystem.mspx ,
12. David G. Messerschmitt and Clemens Szyperski. Software Ecosystem: Understanding an Indispensable Technology and Industry. Cambridge, MA, USA: MIT Press. ISBN 0262134322 – 2003.
13. GEORGE G. MITCHELL and JAMES DECLAN DELANEY. An Assessment Strategy to Determine Learning Outcomes in a Software Engineering Problem-based Learning Course. Int. J. Engng Ed. Vol. 20, No. 3, pp. 494-502, 2004 Printed in Great Britain.
14. Ghazi Alkhatib, Ghassan Issa, Aiman Turani, M. Ibrahim Zaroor. Incorporating Innovative Practices in Software Engineering Education. April 04 - 06, 2011, IEEE DUCON Education Engineering 2011 - Learning Environments and Ecosystems in Engineering Education. Amman, Jordan.
15. Olavo Barbosa, Carina Alves. A systematic mapping study on software ecosystems.
16. Proceedings of the Third International Workshopon Software Ecosystems – IWSECO-2011, Brussels, Belgium. – June 7th, 2011. – Vol.746. – pp. 15 – 26.
17. Jansen, S., Finkelstein, A., Brinkkemper, S.: A sense of community: A research agenda for software ecosystems. In: International Conference on Software Engineering. ICSE-Companion 2009, Vancouver, British Columbia, Canada (2009).
18. Manikas, K., Hansen K.M.: Software ecosystems–a systematic literature review. Journal of Systems and Software 86, 2012. – vol 3. – p. 71–80.
19. Weiblen, T., Giessmann, A., Bonakdar, A., Eisert, U.: Leveraging the software ecosystem: Towards a business model framework for marketplaces. In: 3rd International Conference on Data Communication Networking, DCNET 2012, 7th International Conference on e-Business, ICE-B 2012 and 3rd International Conference.

UDK 004.254:004.052(045)
Melnyk V.M., Pekh P.A., Melnyk K.V., Zhyharevych O.K.
Lutsk national technical university

SIGNIFICANCE OF THE SOCKET PROGRAMMING FOR THE LABORATORY WITH INTENSIVE DATA COMMUNICATIONS

Melnyk V.M., Pekh P.A., Melnyk K.V., Zhyharevych O.K. Significance of the socket programming for the laboratory with intensive data communications. Many courses based on data communications are connecting with no programming content. They are designed for computer science topics and should include programming. A lot of data communication courses with a programming component make use of serial ports on PCs while some of them deal with detailed network layer projects. UNIX socket programming allows the learners to deal with the same issues and problems, but in a context that is more to be useful and interesting. In addition, if classes with sockets are used with C++, only as much detail of socket operation as desired need be presented.
Keywords: data communications, socket programming, contention, C++ socket classes, server.

Мельник В.М., Пех П.А., Мельник К.В. Жигаревич О.К. Важливість сокетного програмування для лабораторій з інтенсивним обміном даних. Багато курсів, засновані на передачі даних, підключаються без програмного контенту. Вони створені для тем з області комп□ютерних наук і повинні включати програмування. Багато курсів на базі комунікації даних з компонентом програмування можуть використовувати послідовні порти на ПК, в той час як деякі з них взаємодіють з конкретними мережевими рівнями проектів. Програмування UNIX-сокетів дозволяє вивчаючим займатися тими ж питаннями і проблемами, але в контексті, що може бути більш корисним і цікавим. Крім того, представлення сокетних класів з використанням C++ повинно настільки деталізувати операції сокетів, скільки це необхідно.
Ключові слова: обмін даними, сокетне програмування, зв□язок, класи сокетів C++, сервер.

Мельник В.М., Пех П.А., Мельник К.В. Жигаревич О.К. Важность сокетного программирования для лабораторий с интенсивным обменом данных. Многие курсы, основанные на передаче данных, подключаются без содержания программирования. Они предназначены для информатики темы и должен включать программного контента. Много курсов на основании коммуникации данных с компонентом программирования могут использовать последовательные порты на ПК, в то время как некоторые из них взаимодействуют с конкретными слоями сетевых проектов. Программирование UNIX-сокетов позволяет учащимся заниматься теми же вопросами и проблемами, но в контексте, который является более полезным и интересным. Кроме того, представление сокетных классов с использованием C++ должно настолько детализировать операции сокетов, сколько это необходимо.
Ключевые слова: обмен данными, сокетное программирование, связь, классы сокетов C++, сервер.

Introduction. Data communications is a common part of most Multi Interface Socket (MIS) and Client Socket (CS) programs. As evidenced, the actual implementation of the course varies widely by the available text books variety. Many texts, oriented on toward MIS or CS, provide little or absolutely no laboratory activity. MIS programs tendency to emphasize on a data communications and networks management. Recent news lists postings indicate an emphasis on using data communications and investigations of the types and available communication styles. National or international cooperative projects are too popular and CS programs may use very technical texts or a broad text, such as used in [1], where principles, design approaches and standards are going to be highlighted. Obviously, an engineering program would have a much more extensive and detailed course/courses to investigate the physical and structural data communication aspects. The course taught by the author requires for all CS majors. Students or outside users in the course maybe in any of the specialized tracks (artificial intelligence, business information systems, graphics or scientific programming) as well as more generic CS major.

The possible laboratory experiences types are also wide-ranging. The ″global cooperation model″ teaches how data communications works by forcing students to use sophisticated communication mechanisms and provides a basis for explaining how these systems function. It is possible to consider design alternatives, based on the available resources, by allowing learners to explore different physical or logical communication types.

At the other extreme are exercises that emphasize low-level physical understanding of data communications almost an engineering approach. A typical example could be the use of serial ports on PCs. In addition to writing code with aim to manipulate the physical hardware can be studied many more complicated concepts. In material the author has used in the past [2], file transfer assignments using a modified BiSynch protocol and implemented token rings. An alternative low-level approach is modelled by the NetCp software [3]. This laboratory tactic involves a large scale project based on the 0SI IS0 data

link layer developing. None of these approaches provide practical hands-on hardware experience. In addition to the exercises described in this work, the author assigns a project involving installation of hardware and software to add a PC to a network. A server can be installed and configured for extra credit. Such a project was continued after the socket model was adopted. Users placed in practice or in their first job to consider how the data communications course is important and they have typically not believed that PC serial port programming is to be so important.

The approach presented here is designed to provide a broad overview of data communication and network issues to students. The goals for the laboratory part of the data communication course also were presented later in publications.

UNIX sockets. Simply say, sockets area mechanism by which messages may be sent between processes on the same or different machines. If the processes are located on the same machine, the sockets may be used as *pipe*s. Internet sockets allow communication between processes running on different machines. The system calls are the same as file input/output. A typical attitude to socket programming is to create a process that opens a server socket port and listens for another process to attempt connection. A client can open a socket with the same port number as the server socket, requesting connection to the service. A connection is established when server hears the request from the client. Communication can now operate using the read() and write() system calls.

There are many types of standard protocols. Two of the most common are UDP (user datagram protocol) and TCP (transmission control protocol). Both protocols transmit packages of information between processes via socket. UDP does not put a guarantee, that data will be received or that a transmission of multiple packets will be received in order. TCP is a stream protocol that is reliable and sequenced. Practically, input and output to the programmer on a TCP socket appears as a byte stream from a terminal or a file. If TCP data cannot be successfully transmitted within a reasonable amount of time, then will be indicated an error. There is less overhead involved in UDP, but programming must be much more sophisticated if there is important orderly message receipt.

The socket connection between two processes usually is a connection between host-port pairs where the port number indicates a particular available service that is made. Many of the services commonly available via TCP sockets are recognizable acronyms: SMTP (Simple Mail Transport Protocol), NNTP (Network News Transport Protocol) and FIT (File Transport Protocol). *Telnet* and *rsh* are also additional socket services. UNIX provides a mechanism whereby the name of an available service is translated to a port number. Sockets are also used for the interprocess communication necessary in concurrent or parallel processing. Therefore, parallel processing assignments as well as data communications projects can be built on the same framework.

Advantages of sockets. One obvious disadvantage of using socket programming for the data communications lab is that they are less direct hardware interaction than with PC serial ports. However, most graduates will not be in situations where such detailed knowledge will be important. Even with the serial port approach the concepts have remained to some extent abstract to many students. The socket based approach has the advantage that the abstract sockets concepts (and practical uses such as mail, telnet, etc.) become much more concrete.

One advantage that PC based labs have had in the past is that they were inexpensive. However, there are at least two factors that balance this advantage. One is that UNIX workstations are now commonly available. PC labs can be converted to workstations by installing free UNIX versions. Most of socket assignments, given in a data communications course, are not compute intensively and do not require a graphical interface; workstations have not to be dedicated to the course as it would be true for PCs. Another factor is that even though PCs are relatively inexpensive, what happens practically is that older, less reliable, machines are assigned to a dedicated project such as a data communications lab. Our experience says that the machines we could use were quite unreliable.

Although the higher level nature of socket programming has been stressed as an advantage. It is possible to make assignments so much detailed as desired. Socket programming without any support software can require a great deal of low level understanding and manipulation. One simple modification

would be to base assignments on UDP packets rather than TCP one. Much additional programming (error checking via CRCS, sequence numbers, acknowledgment of receipt, negative acknowledgment for receipt of a bad packet) would be also necessary. With either UDP or TCP packets, properly designed handshaking mechanisms may be necessary for applications like file transfer.

With serial port assignments, lecture time was devoted to such low level concepts as control, status and data registers, and parallel to serial conversion. With a socket based approach can be discussed analogous concepts such as packet headers and network and machine byte order. If looked-for, many of the topics appropriate for serial port communication can be required for socket programs and many of the same assignments can be given. Even if high level applications are assigned, the learners must still understand the differences between streams and buffers.

Advantages of C++ socket classes. Many references provide details of socket communication [4,5,6]. These references provide examples and ideas for assignments. All of the establishing communications details, converting the communication into a buffered stream and error checking, can be done with UNIX system calls. Much low level understanding may be required to write applications that are stable. A well designed set of C++ classes can be constructed which will provide the full power of sockets with simple semantics requiring. It is possible to write up clients to established servers, event driven servers, polling servers, etc.

The author [7] provided the students with a set of C++ socket classes written and copyright by G. Swaminathan from Electrical Engineering Department. These routines have been written to work with GNU *libg++* and appear the same as the *iostream* library. These classes have functioned very well for the given assignments. The interface is the same as the *iostream* library and provides type-safe input and output. There are *sockstream* classes in the UDP and TCP domains as well as a *pipestream* class. The *sockbuf* classes are derived from the *streambuf* class of the *libg++ iostream* library. Thus, learners must learn about streams and buffers for non-socket input and output.

Sockbuf classes include error functions: ready checks, flush operations, and overflow, underflow, and *timeout* functions. There can also be set socket options, such as message routing, reuse of local address, broadcast etc. Therefore, socket detail may be included as it may be desired.

In this particular prospectus a side benefit of using these C++ classes is that clients (students) are required to use C++ in a junior/senior level course to help them retain skills gained at the freshman/sophomore level.

Assignments. In choosing assignments to give hour course (during a 1,5 year), several goals were desired. It was hoped to design a set of assignments which would require the student to write a client application, a server application a peer-to-peer application and also provide experience with some standard applications such as electronic mail and file transfer. In addition, the assignments should begin simply and become more complicated during the semester. They outlined below met these criteria.

The assignments were very well received by the students. They were perceived to be of practical interest and, at the same time, to be fun projects. Some of them, who don't have a history of applying themselves well to project assignments spent much effort on these assignments and produced good results. Specifically, we are giving five assignments here:

Assignment 1 □ socket client to SMTP server. Write a client program to connect with an SMTT server on a local or remote machine and send a mail message to a *userid*. The user need not be on either the local or remote machine. For example, the program might be named *smtp* and have two arguments: *hostname* and *userid*. A simple command line interface is required but learners were free to develop much more elaborate e-mail style interfaces. There must be used commands understood by SMTP. A subset are follows:

HELO `localname`	Identifies connecting machine □ `localname` is not needed □ some servers do not need HELO, but include it before do something.
HELP	Sends commands list.
MAIL FROM `name`	May be anything you wish □ it is not checked for validity.

RCPT TO: `name`	Recipient of mail □ need not be local `name`.
DATA	Allows entry of message □ terminate message with □.□ as only character on line.
QUIT	Disconnect.

This assignment, as well as others, teaches students how to do improper activities. The following warning was provided to them: *«Obviously one could do ill-mannered things with this program. For example, it could be sent a bunch of messages from Dafi Duck to some administrator. It would take some work, but the sender of these messages could be traced. Please, do not involve in such juvenile behavior»*. Some of them would argue that such assignments are too «dangerous». However, the students are learning how to manipulate sockets and could figure out how to send such mail on their own. For this it need to be ready to acknowledge the problem and announce a warning.

Assignment 2 □ simple network information server. Write a network server program which will behave as follows: accept commands from the input socket; interpret the commands and gather the information; and send the commands output to the output socket.

You will no need to write a client program for this assignment as the standard telnet client will provide the necessary functions. Telnet allows you to send information over a client to a server process and handles the return information printing. A selection of information providing system commands such as domain name, who, etc. was chosen. The system functions can be executed from within a C++ program. The difficult part is to take the commands output and send the output to the socket connected to the client. The output of the commands should be connected directly to the socket. Two approaches are suggested: using the *pipe*stream class and using a traditional C fork() to execute the system function which is connected by a user-constructed *pipe*.

Assignment 3 – peer-to-peer socket communications. Write a "chat" program which will execute as two idential programs. It should allow users to type information that will appear as output on the connected process. The two processes have to be connected via socket. The program will allow the user to connect to a certain process or to listen for another process or trying to connect to it.

The same program runs on both machines. Topics necessary for this assignment include: timeout on listen, creating a child process as done by many server programs, closing sockets and killing child processes. A finite-state transition model could be presented to help in this program design.

Assignment 4 □ file transfer □ client and server. Write a pair or programs to transfer files over a TCP/IP network socket connection. The first program should operate in much the same way as an FIT server. It should be run in the background and wait for a connection on a specified port. The second program will function in much the same way as an FIT client. Therefore, a user interface will be needed. Commands will be entered and sent to the server with noted responses and files may be transferred in either direction. The client program should accept the following commands with corresponding actions:

`ls`	list files on server
`put`	transfer file from client to server
`get`	transfer file from server to client
`quit`	disconnect from server
`:<command>`	execute `<command>` on client useful for `ls` on client

The capabilities (and therefore, protocol) of this client/server pair are much simpler than FIT. SFTP (Simple File Transfer Protocol) is closer to what is required. FTP, for example, uses 2 TCP connections: a telnet-like connection for control and a second one for data transfer. SFTP uses a single TCP connection and supports user access control, directory listing and changing, file renaming and tile erasure. There, only directory listing is required here for these commands. FIT also supports *lcd*, input, message*get*, etc. A hand-shaking protocol was required for this assignment.

Assignment 5 – three options were given:

Assignment 5A – FIT file transfer using UDP. Implement the file transfer programs of assignment 4 built on UDP sockets rather than TCP sockets. The programs will need to assemble data packets, provide CRC error checking and provide packet sequencing.

Packets may arrive in different order, duplicated or missing. They may need to be re-requested and/or rearranged. Each packet should be acknowledged (positively or negatively). A protocol will need to be adopted, which will describe the packet format error messages, acknowledgments, etc. In order to test the robustness of the protocol used, allow the user to specify transmissions fractions that will (randomly) be done in error.

Assignment 5B – two channel bidirectional file transfer. Implement of the file transfer programs of assignment 4 modified to have two sockets open: one for control information and other for data transfer. In addition, it need to allow the two programs to send files back and forth at the same time. A transfer can be aborted by using the control channel. It may be helpful to fork multiple processes (a finite state machine approach would be a good idea). FTP works in a similar manner. It has two socket connections, but does for different reasons since it is a true client-server protocol rather than the peer-to-peer protocol implemented here.

Assignment 5C – multi-user chat program. Assignment 2 involved a peer-to-peer chat program. This assignment requires a multiplexing chat server program creation which can handle multiple socket connections. There is no need to write a client program because telnet can be used here.

The server should accept input lines from any connected socket and output them to the rest of the connected sockets. When a user connects to the chat server, the server should prompt for a user name. This name should be broadcast to the rest of the users as ″user <name> has just joined the session″. Similarly, a message should be broadcast when the user leaves the conference. When a user's message is sent to the other connected users, the user name should be included for identification.

Conclusions. The goals of the laboratory component redesigning of the data communications course were to provide assignments that are: more meaningful and practical to the students; more enjoyable and, therefore, will be done better; more modern but are still oriented toward understanding, what is happening rather than simply using data communications; increasingly complex and build on previous assignments; based on more reliable hardware that the discarded PCs used previously.

Once the socket paradigm was chosen, goals were to create assignments which required students to write code that: makes use of C++ classes; provides simple client access to a well-defined server; provides simple server functionality; provides peer-to-peer communication; provides multiplexing server functionality; functions in a manner similar to a well-known network service; requires students to be concerned with unreliable communications; uses some form of fork() and programming for interprocess communications. The use of C++ socket classes and the assignments chosen meet all of the above goals if the optional forms of assignment 5 are chosen.

1. W. Stallings. Data and Computer Communications. / 4-th edition, MacMillan, New York, 1994.
2. W. D. Smith. The Design of An Inexpensive Undergraduate Data Communications Laboratory, SIGC'SE Bulletin. 1991. Vol 23, № 1. p. 273-276.
3. D. Finkel, S. Chandra, NetCp A Project Environment for an Undergraduate Computer Networks Course, SIGCSE Bulletin. 1994. Vol 26, № 1. p. 174-177.
4. A Socket-Based Interprocess Communications Tutorial. Chapter 10 of SunOS Network Programming Guide.
5. An Advanced Socket-Based InterProcess Communications Tutorial. Chapter 11 of SunOS Network Programming Guide.
6. R. Quinton. An Introduction to Socket Programming, Computing and Communication Skrvices. The University of Western Ontario, London, Ontario.
7. G. Swaminathan. C++ Socket Classes, Electrical Engineering Department, University of Virginia.

УДК 004.773:614.29 (045)
Мельник В.М., Пех П.А., Мельник М.М.
Луцький національний технічний університет

LINUX-КОНТЕЙНЕРИ І МАЙБУТНІЙ CLOUD

Melnyk V., Pekh P., Melnyk M. Linux containers and the future cloud. Linux-based containers are described in this article, and briefly, explanations of the underlying c-groups and namespaces kernel features are put in the article. Some Linux-based container projects are also discussing here, focusing on the promising and popular LXC (LinuX Containers) project. There also is a look at the LXC-based Docker engine, which provides an easy and convenient way to create and deploy LXC containers. Several hands-on examples showed how simple it is to configure, manage and deploy LXC containers with the userspace LXC tools and the Docker tools.

Due to the advantages of the LXC and the Docker open-source projects, and due to the convenient and simple tools to create, deploy and configure LXC containers, as described in this article, we presumably will see more and more cloud infrastructures that will integrate LXC containers instead of using virtual machines in the near future. However, as explained in this article, solutions like Xen or KVM [1] have several advantages over Linux-based containers and still are in need, so they probably will not disappear from the cloud infrastructure in the next few years.

Keywords: Linux-based container, kernel features, Docker engine, userspace LXC tools, Docker tools, cloud infrastructures

Мельник В.М., Пех П.А., Мельник М.М. Linux-контейнери і майбутній cloud. В роботі описані контейнери на базі операційної системи Linux, а також поданий детальний опис ведучих С-груп та можливості просторів імен ядра. Деякі контейнерні проекти на базі Linux також обговорюються з погляду їх перспектив застосування та актуальності в LXC проектах. Також обговорюється погляд на Docker engine на базі LXC, який забезпечує легкий та зручний шлях для створення і розширення LXC-контейнерів. Багато створених зручних варіантів розробок довели, як просто конфігурувати, управляти і розширювати LXC-контейнери засобами LXC простору користувача і засобами Docker.

В зв'язку з перевагами LXC- і Docker-проектів з доступними ресурсами і в зв'язку з зручністю та простотою засобів створення, поширення та конфігурації LXC-контейнерів, як описано в цій роботі, з наполегливістю більше і більше розглядатимуться cloud-інфраструктури, які в близькому майбутньому здійснюватимуть інтеграцію LXC-контейнерів замість використання віртуальних машин. Однак, як пояснюється в даній статті, висновки подібні Xen чи KVM [1] мають деякі переваги над базовими Linux-контейнерами і, все ж, є потрібними, а тому не можуть просто зникнути з cloud-інфраструктури протягом декількох наступних років.

Ключові слова: контейнери на базі операційної системи Linux, можливості ядра, Docker engine, засоби LXC простору користувача, засобами Docker, cloud-інфраструктури

Мельник В.М., Пех П.А., Мельник М.М. Linux-контейнеры і будущий cloud. В работе описаны контейнеры на базе операционной системы Linux, а также подано детальное описание основных С-групп и возможности пространств имен ядра. Некоторые контейнерные проекты на базе Linux также обсуждаются с точки зрения их перспектив использования и актуальности в LXC проектах. Также обсуждается взгляд на Docker engine на базе LXC, который обеспечивает легкий та выгодный путь для создания и расширения LXC-контейнеров. Много созданных удобных вариантов разработок довели, как просто конфигурировать, управлять и расширять LXC-контейнеры посредствами LXC пространства пользователя і средствами Docker.

В связи с преимуществами LXC- і Docker-проектов с открытыми ресурсами и в связи с удобностью и простотой средств создания, расширения та конфигурации LXC-контейнеров, как описано в этой роботе, з намерением больше и больше рассматриваются cloud-инфраструктуры, которые в близком будущем будут производить интеграцию LXC-контейнеров вместо использования виртуальных машин. Все же, как объясняется в данной статье, выводы подобны Xen или KVM [1] имеют некоторые преимущества над базовыми Linux-контейнерами и, все ж, есть нужными, а поэтому не могут просто исчезнуть з cloud-инфраструктуры на протяжении нескольких следующих лет.

Ключевые слова: контейнеры на базе операционной системы Linux, возможности ядра, Docker engine, средства LXC пространства пользователя, средства Docker, cloud-инфраструктуры

Постановка проблеми. Контейнерна інфраструктура на базі Linux являє собою cloud-технологію, засновану на швидкій і зручній віртуалізації процесів [1]. Вона забезпечує своїм користувачам середовище найбільш близьке до стандартного Linux дистрибутива. В протилежність рішенням пара-віртуалізації (Xen) та рішенням апаратної віртуалізації (KVM), які підтримуються віртуальними машинами (ВМ), контейнери не створюють екземплярів ядра операційних систем. У зв'язку з тим фактом, що контейнери є легше застосовні, ніж ВМ, то можна домогтися більш високого зближення Linux з контейнерами, ніж з віртуальними машинами на одному і тому ж хості. З практичної точки зору, можна розгорнути більше примірників контейнерів, ніж віртуальних машин на одному і тому ж хості.

Іншою перевагою контейнерів над ВМ є те, що запуск і зупинка контейнера здійснюється набагато швидше, ніж запуск і вимикання ВМ. Всі контейнери для хоста виконуються під тим же ядром, на відміну від рішень для віртуалізації, таких як Xen або KVM, де кожна віртуальна машина володіє власним ядром. Іноді обмеження виконання під тим же ядром у всіх контейнерах одного і того ж хоста можуть вважатися недоліком. Крім того, не можна запустити BSD, Solaris, OS/X або Windows, в контейнері на базі Linux, а іноді і цей факт також можна вважати недоліком [1,4].

Ідея віртуалізації на рівні процесів саме по собі не нова, а вже була реалізована за допомогою зон Solaris так як і пов'язаних BSD вже кілька років тому. Інші проекти з відкритим вихідним кодом впровадження віртуалізації на рівні процесів існували протягом кількох років. Однак, вони вимагали ядер використання, які часто ставали основною перешкодою. Повна і стабільна підтримка для контейнерів на базі Linux для звичайних ядер в рамках проекту LXC (LinuX Containers) появилася відносно недавно, як можна буде побачити в даній статті. Це робить контейнери більш привабливим для cloud-інфраструктури. Все більше і більше компаній хостингових і cloud-послуг приймають і дотримуються ідей контейнерів на базі Linux. У цій статті *ставиться за мету* описати деякі контейнерні проекти з відкритим кодом на базі Linux і функції ядра, які вони використовують, а також продемонструвати деякі приклади використання. *Метою* також буде опис інструменту Docker для створення LXC контейнерів [4].

Базова інфраструктура сучасних контейнерів на базі Linux складається в основному з двох властивостей ядра: просторів імен і С-груп. Є шість типів просторів імен, які забезпечують для кожного процесу ізоляцію наступних ресурсів операційної системи: файлових систем (MNT), UTS, IPC, PID, просторів імен мережі і користувача (простори імен користувачів дозволяють відображення UID і GID між простором імен користувача і глобальним простором імен хоста). Використовуючи мережеві простори імен, наприклад, кожен процес може мати власний примірник мережевого стека (мережевих інтерфейсів, сокетів, таблиць маршрутизації і правил маршрутизації, правил мережевого фільтра (Netfilter) і так далі).

Створення мережевого простору імен дуже просте і може бути зроблено за допомогою наступної команди `ip route`: `ip netns add myns1`. За допомогою команди `ip netns` також легко переміщати один мережевий інтерфейс з одного мережевого простору імен до іншого для контролю створення і видалення мережевих просторів імен, з метою з'ясувати, якому мережевому простору імен належить визначений процес і так далі. Цілком аналогічно, при використанні простору імен MNT під час монтування файлової системи інші процеси не будуть бачити це монтування, а уже при роботі з просторами імен PID, запустивши на виконання команду ps від даного простору імен PID можна побачити тільки процеси, які були створені виходячи саме з цього простору імен PID.

Підсистема С-груп забезпечує управління ресурсами та їх облік. Це дозволяє легко визначити, наприклад, максимальний обсяг пам'яті використання процесом і може реалізуватися за допомогою VFS-операцій С-груп. Проект С-груп було вперше розпочато двома розробниками Google, Павлом Мінеджі і Рогітом Сетом ще в 2006 році, і він спочатку називався "контейнери процесу" [1]. Ні простори імен, ні С-групи не брали участі в критичних шляхах ядра, і, таким чином, вони не містять значних негативних показників високої продуктивності, за винятком пам'яті для С-групи, яка може включати значні затрати при деяких робочих навантаженнях.

Контейнери на базі Linux

В принципі, контейнер являє собою Linux процес (або декілька процесів), що має свої особливості і працює в ізольованому середовищі, налаштованому на хості. Можна іноді стикатися з такими термінами, як віртуальне середовище (ВС або virtual environment VE) і Віртуальний власний сервер (ВВС або Virtual Private Server VPS) для контейнера.

Особливості цього контейнера залежить від того, як контейнер сконфігурований і на якому базовому Linux-контейнері використовується, так як контейнери Linux реалізовані по-різному в різних проектах. Це обговорюється в найбільш важливих публікаціях.

Походження проекту OpenVZ [2] є власним варіантом реалізації серверів віртуалізації під назвою Virtuozzo, який спочатку був запущений компанією під назвою SWsoft, заснованою в 1997 році. У 2005 році частина продукту Virtuozzo була випущена в якості відкритого вихідного проекту, і він називався OpenVZ. В 2008 році SWsoft об□єдналася з компанією Parallels. OpenVZ

використовується для надання хостингу і cloud-сервісів, що являється основою спаралелених cloud-серверів. Як Virtuozzo так і OpenVZ базується на модифікованому ядрі Linux. До того ж, у нього є інструменти командного рядка для управління контейнерів, і це дає можливість створювати контейнери з використанням шаблонів для різних дистрибутивів Linux. OpenVZ також може працювати на деяких немодифікованих ядрах, однак зі зменшеним набором функцій. Проект OpenVZ призначений для повноцінного впровадження в майбутньому, але це може зайняти досить багато часу.

В 2013 році компанія Google випустила версію свого власного контейнерного стека з відкритим вихідним кодом lmctfy (що розшифровується як Let Me Contain That For You). На даний час це все ще в стадії тестування. Проект lmctfy заснований на використанні C-груп. На даний час контейнери Google не використовують особливості простору імен ядра, що використовується в інших контейнерних проектах на базі Linux, але використання цієї особливості для контейнерного проекту Google очікується в майбутньому [1].

Проект з відкритим вихідним кодом [3], який був вперше реалізований в 2001 році, забезпечує можливість для безпечного розподілу ресурсів на хості, який, в свою чергу, повинен запустити модифіковане ядро.

Проект LXC (LinuX Containers) надає користувачу набір інструментів і утиліт для управління контейнерами Linux [4]. Багато LXC розробників вийшли з OpenVZ групи. В супереч OpenVZ він працює на немодефікованому ядрі. LXC повністю написаний для контексту користувача і підтримує прив□язки на інших мовах програмування, таких як Python, Lua і Go. Він доступний для більшості популярних дистрибутивів, таких як Fedora, Ubuntu, Debian та інших. Red Hat Enterprise Linux 6 (RHEL6) представив контейнери Linux в якості технічного попереднього перегляду. Можна запустити Linux контейнери на архітектурах, інших ніж x86, таких як ARM архітектури (є кілька прикладів в Інтернеті для запуску контейнерів на Raspberry Pi).

Можна також згадати драйвер Libvirt-LXC, за допомогою якого можна управляти контейнерами [5]. Це робиться шляхом визначення конфігураційних файлів XML, а потім працює `virsh start`, `virsh console` і `visrh destroy` для роботи, розміщення і знищення контейнера, відповідно. Зверніть увагу, що немає єдиного коду між Libvirt-LXC та LXC-проекту користувача.

Управління контейнерами LXC

По-перше, слід переконатися, що хост підтримує LXC, запустивши `LXC-checkconfig` [4]. Якщо все гаразд, можна створити контейнер за допомогою одного з декількох готових шаблонів для створення контейнерів. У LXC-0,9 є таких шаблонів, в основному для популярних дистрибутивів Linux. Можна легко адаптувати ці шаблони відповідно до власних вимог при необхідності. Так, наприклад, можна створити контейнер Fedora під назвою fedoraCT за допомогою команди:

```
lxc-create -t fedora -n fedoraCT
```

Контейнер буде створений за замовчуванням, в `/var/lib/lxc/fedoraCT`. Можна встановити інший шлях для згенерованого контейнера, додавши `--lxcpath` опцію шляху. Опція `-t` визначає ім□я шаблону, який необхідно використовувати (в даному випадку Fedora), і опція `-n` вказує на ім□я контейнера (в даному випадку `fedoraCT`). Слід звернути увагу на те, що можна також створити контейнери інших розподілів на Fedora, наприклад Ubuntu (для нього необхідний пакет Debootstrap). Не всі також комбінації гарантується.

Можна передавати параметри в `lxc-create` після додавання `--`. Наприклад, можна створити старшу версію для кількох дистрибутивів з опціями `-R`, або `-r`, залежно від шаблону дистрибутиву. Для створення старшої версії контейнера Fedora на тому ж хості, Fedora 20, слід вжити команду:

```
lxc-create -t fedora -n fedora19 -- -R 19
```

Можна видалити установку LXC-контейнера з файлової системи за допомогою:

```
lxc-destroy -n fedoraCT
```

Для більшості шаблонів, коли вони використовуються вперше, декілька файлів необхідного пакету завантажуються і кешуються на диску в `/var/cache/lxc`. Ці файли використовуються

при створенні нового контейнера з тим же самим шаблоном, і, в результаті, створення контейнера, який використовує цей же шаблон, буде швидшим в наступному. Можна почати контейнер, який був створений:

```
lxc-start -n fedoraCT
```

і зупинити з:

```
lxc-stop -n fedoraCT
```

Сигнал, що використовувався `lxc-stop` є SIGPWR за замовчуванням. Для того, щоб використовувати SIGKILL в попередньому прикладі, слід додати `-k` для `lxc-stop`:

```
lxc-stop -n fedoraCT -k
```

Можна також створити контейнер в якості демона, додавши `-d`, а потім увійти в нього з `lxc-console`, як це показано нижче:

```
lxc-start -d -n fedoraCT lxc-console -n fedoraCT
```

Перший `lxc-console`, який можна запустити для даного контейнера зв□яжеться з `tty1`. Якщо `tty1` вже використовується (бо це друга LXC-консоль, яка запущена для цього контейнера), то відбудеться з□єднання з `tty2` і так далі. Слід мати на увазі, що максимальна кількість `tty□s` налаштовується на початку `lxc.tty` конфігураційного файлу контейнера. Можна зробити миттєвий знімок незапущеного контейнера через:

```
lxc-snapshot -n fedoraCT
```

Це дозволить створити миттєвий знімок під `/var/lib/lxcsnaps/fedoraCT`. Перший знімок, що буде створений, буде називатися `snap0`. Другий □ буде називатися `snap1` і так далі. Можна відтворити в часі миттєвий знімок на пізніший час з опцією `-r` □ наприклад:

```
lxc-snapshot -n fedoraCT -r snap0 restoredFdoraCT
```

Можна перерахувати знімки, використовуючи:

```
lxc-snapshot -L -n fedoraCT
```

Можна відобразити запущені контейнери використавши:

```
lxc-ls --active
```

Контейнери управління також можуть бути зроблені за допомогою скриптів, з використанням мов сценаріїв. Наприклад, цей короткий скрипт *Python* запускає контейнер *fedoraCT*:

```
#!/usr/bin/python3 import lxc container = lxc.Container ("fedoraCT")
                   container.start()
```

Конфігурація контейнерів

Файл конфігурації за замовчуванням створюється для кожного новоствореного контейнера в `/var/lib/lxc/<containerName>/config`, але можна змінити це за допомогою опції задавання місця знаходження `--lxcpath` [5]. Можна також налаштувати різні параметри контейнера, наприклад, параметри мережі, параметри С-груп, пристроїв та багато іншого. Наведемо деякі приклади популярних елементів конфігурації для файла конфігурації контейнера:

– Можете встановити різні параметри С-груп шляхом установки значень у входи `lxc.c-group.[subsystem name]` у файлі конфігурації. Ім□я підсистеми є іменем контролера С-групи. Наприклад, конфігуруючи максимальний обсяг пам□яті 256 MB, які контейнер може використовувати, можна через встановлення властивості `lxc.c-group.memory.limit_in_bytes` значення 256.

– Можна налаштувати контейнер хоста, встановивши `lxc.utsname`.

– Є п□ять типів мережевих інтерфейсів, які можна встановити за допомогою параметра `lxc.network.type`: `empty`, `veth`, `vlan`, `macvlan` і `phys`.`veth` дуже часто використовується для можливості підключення контейнера із зовнішнім світом. При використанні `phys` можна переміщати мережеві інтерфейси від мережевого простору імен вузла в простір імен контейнера.

– Існують можливості, які можуть бути використані для покращення безпеки LXC контейнерів. Можна уникнути реалізації деяких зазначених системних викликів зсередини контейнера, встановивши захищений режим обчислень, або `seccomp`, політику входу через `lxc.seccomp` у файлі конфігурації. Також можна відключити ці можливості з контейнера через вхід `lxc.cap.drop`. Наприклад, встановлення `lxc.cap.drop = sys_module` створить контейнер

без можливості `CAP_SYS_MDOULE`. Спроба запустити insmod зсередини цього контейнера не вдасться. Також для такого контейнера можна визначити профілі `AppArmor` і `SELinux`, а також знайти приклади в `LXC README` і в `man 5 lxc.conf`.

Докер

Docker – це проект з відкритим вихідним кодом, який автоматизує створення і розгортання контейнерів [6]. Перший docker був випущений в березні 2013 року, ліцензія Apache версії 2.0. Вона стартувала в той час як внутрішній проект платформенно-сервісної (Platform-as-a-Service або PaaS) компанії під назвою dotCloud, і тепер називається Docker Inc. Вихідний прототип був написаний в Python, а пізніше весь проект був переписаний в мовою програмування Go, яка була розроблена вперше в Google. У вересні 2013 року, Red Hat оголосила, що вона буде співпрацювати з Docker Inc для Red Hat Enterprise Linux і для платформи Red Hat OpenShift. Для Docker потрібне ядро Linux 3.8 (або вище). У системах RHEL Docker працює під ядро 2.6.32, так як і були перенесені необхідні виправлення.

Docker використовує інструментарій LXC який є в даний час доступний тільки для Linux. Він працює на дистрибутивах, таких як Ubuntu 12.04, 13.04; Fedora 19 і 20; RHEL 6.5 і вищих; і на cloud-платформах, таких як Amazon EC2, Google Compute Engine і Rackspace.

Зображення Docker може зберігатись в загальнодоступному репозиторії і може бути завантажене за допомогою команди `docker pull`, наприклад, `docker pull ubuntu` або `docker pull busybox`. Для відображення зображень наявних на хості можна використовувати команду `docker images`. Можна звузити команду для певного типу зображень (наприклад, Fedora) за допомогою `docker images fedora`. На Fedora, працюючи з Docker-контейнером Fedora, все простіше. Після установки `docker-io package` можна просто запустити `docker dæmon` з `systemctl start docker`, а потім можна стартувати Docker контейнер Fedora як `docker run -i -t fedora /bin/bash`.

Docker має *git*-подібні можливості для обробки контейнерів. Зміни, внесені в контейнері, будуть втрачені, якщо контейнер знищиться і вони не зафіксуються (так само, як робиться в *git*) з `docker commit <containerId> <containerName/containerTag>`. Ці зображення можуть бути завантажені на доступний регістр, і відкриті для бажаючих завантажити їх. Крім того, можна встановити власне ззовні недоступнене сховище Docker.

Docker здатний створити знімок за допомогою можливостей пристрою зображення ядра. У більш ранніх версіях до версії Докер 0.7 це було зроблено за допомогою AUFS (union filesystem). Docker 0.7 складає "плагіни зберігання" для того, щоб при необхідності можна було перемикатися між відображеннями пристроїв і AUFS (якщо ядро підтримує його). Таким чином Docker може працювати на релізи RHEL, які не підтримують AUFS.

Можна створювати зображення, виконуючи команди вручну і створювати в результаті контейнер, але також можна описати їх з Dockerfile. Так само як Makefile буде компілювати код в двійковий файл виконання, Dockerfile будуватиме готове до запуску контейнерне зображення виходячи з простих інструкцій. Команда для створення зображення з Dockerfile є `docker build`. Існує підручник про Dockerfiles і їх синтаксис команд на веб-сайті Docker. Наприклад, після короткого Dockerfile для установки пакета `iperf` для зображення Fedora:

```
FROM fedora MAINTAINER Rami Rosen RUN yum install -y iperf
```

Можна завантажувати і зберігати зображення безкоштовно на відкритому індексі Докера. Так само, як і у випадку з GitHub, зберігання доступних зображень безкоштовне і просто вимагає, щоб користувач зареєструвався.

Можливість КТВ

Проект КТВ (контрольна точка / відновлення) реалізується в основному в просторі користувача і існує більш ніж 100 маленьких ″ділянок″, розміщені по ядру для його підтримки. Було кілька спроб реалізації КТВ окремо в просторі ядра, а деякі з них – в рамках проекту OpenVZ. Спільнота ядра відхилила всі з них через те, що вони були занадто складні [7].

Можливість КТВ дозволяє зберегти стан процесу в декількох файлах зображень і відновити цей процес з точки, в якій він був остановлений, на тому ж хості або на іншому пізніше в часі. Цей процес також може бути контейнером LXC. Файли зображень створюються з використанням

формату буфера протоколу (БП) Google. Можливість КТВ дозволяє виконання завдань, таких як оновлення технічного обслуговування ядра або апаратного обладнання на цьому ж хості після моменту перевірки його додатків для постійного зберігання. Пізніше, додатки будуть відновлені на цьому хості.

Ще одною дуже важливою особливістю є балансування навантаження за допомогою динамічної міграції. Можливості КТ/В також можуть бути використані для створення інкрементних знімків, які можуть бути використані після виникнення аварії. Як уже згадувалося раніше, деякі "ділянки" ядра були необхідні для підтримки КТ/В в просторі користувача. Наведемо деякі з них:

- Був доданий новий системний виклик `kcmp()` який порівнює два процеси, щоб визначити, чи вони займають ресурс ядра.
- Інтерфейс сокетного моніторингу `sock_diag` був доданий в сокети UNIX щоб бути в змозі знайти знаходження сокета UNIX домена. Перед цією зміною інструмент `ss`, який спирався на аналіз записів `/proc`, не виявив цю інформацію.
- Був доданий режим налагодження TCP з єднання.
- Був доданий вхід `procfs` (/proc/PID/map_files).
- Давайте подивимося на простий приклад використання функції `criu` (КТ/В в просторі користувача). По-перше, необхідно перевірити, чи підтримує ядро КТ/В, запустивши `criu check --ms`. Слід завбачити відповідь, яка говорить `"Looks good."`

В основному, визначена перевірка здійснюється через:

```
criu dump -t <pid>
```

Можна вказати папку, в якій наявні файли процесу будуть збережені, додавши `-D folderName` і можна відновити їх з використанням `criu restore <pid>`.

Висновки

У даній роботі описано контейнери на базі Linux і коротко пояснено, що в основі лежать С-групи і простір імен функцій ядра. Обговорено кілька проектів з використанням контейнерів на базі Linux, з концентрацією уваги на перспективний і популярний проект LXC. Також в роботі приділено увагу `Docker engine` наоснові LXC, який забезпечує легкий і зручний спосіб для створення і розширення LXC-контейнерів. На прикладі кількох наочних прикладів показано, як просто конфігурувати LXC-контейнери, керувати ними і розгортати їх з використанням засобів LXC простору користувача та засобів Docker.

Використовуючи переваги LXC- і Docker-проектів з відкритим кодом та зручні і прості інструменти для створення, розгортання і налаштування LXC-контейнерів, як описано в цій статті, можна буде, ймовірно, побачити більше і більше cloud-інфраструктур, які будуть інтегрувати LXC контейнери замість використання віртуальних машин в найближчому майбутньому. Тим не менш, як наголошується в цій роботі, рішення, такі як Xen або KVM мають ряд переваг перед контейнерами на базі Linux і все ж є необхідними, так що вони, ймовірно, не можуть віджити з cloud-інфраструктури в найближчі кілька років.

1. Google Containers: https://github.com/google/lmctfy
2. OpenVZ: http://openvz.org/Main_Page
3. Linux-VServer: http://linux-vserver.org/
4. LXC: http://linuxcontainers.org/
5. libvirt-lxc: http://libvirt.org/drvlxc.html
6. Docker: https://www.docker.io/
7. Docker Public Registry: https://index.docker.io/

УДК 519.687.7
Плахотний М.В. доц. каф. СП та СКС ФПМ НТУУ КПІ
Наливайчук М.В. ст. викл. каф. СП та СКС ФПМ НТУУ КПІ
Івасюк В.М. магістр каф. СП та СКС ФПМ НТУУ КПІ

ВИКОРИСТАННЯ ПЛАТФОРМИ DEVICEHIVE DISCOVERY RPI В НАВЧАЛЬНОМУ ПРОЦЕСІ

Плахотний М.В. Наливайчук М.В., Івасюк В.М. Використання платформи DeviceHive Discovery RPi в навчальному процесі. В статті розглядаються медоди використання платформи DeviceHive Discovery RPi в навчальному процесі. Розглянуто особливості роботи з платформою, та сформульовані задачі при вивченні роботи з периферійними елементами макета.
Ключові слова: DeviceHive, шина, процесор, память, реле, світлодіод, ZigBee.

Ткачук Н.М. Использование платформы DeviceHive Discovery RPi в учебном процессе. В статье рассматриваются медоды использования платформы DeviceHive Discovery RPi в учебном процессе. Рассмотрены особенности работы с платформой, и сформулированы задачи при изучении работы с периферийными элементами макета.
Ключевые слова: DeviceHive, шина, процессор, память, реле, светодиод, ZigBee..

Plahotnyi M. V., Nalyvaichuk M. V., Ivasjuk V. M. Using the platform DeviceHive Discovery RPi in the educational process. This article discusses the use of the platform Stuff DeviceHive Discovery RPi in the learning process. The features of the work platform, and formulated the tasks in the study with the layout of peripheral elements.
Keywords: DeviceHive, bus, processor, memory, relays, LED, ZigBee ...

Постановка наукової проблеми. Сучасна індустрія виробництва багатофункціональних мікроконтролерних засобів дозволяє використовувати їх в навчальному процесі. В статті досліджується можливість використання апаратно-програмної платформи **DEVICEHIVE DISCOVERY RPI** для розробки [1], починаючи від локальних пристроїв до складних інформаційно управляючих систем – наприклад інтелектуальний дім .

Апартно програмні засоби. Загальний вигляд платформи **DEVICE HIVE ARDUINO DISCAVERY** приведено на рис.1.

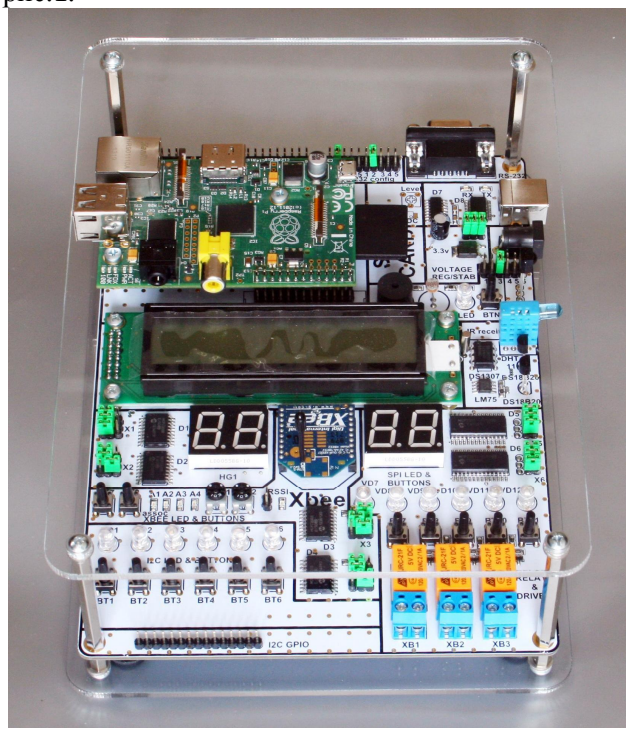


Рис1. Платформа **DEVICE HIVE ARDUINO DISCAVERY**

Основою платформи є плата Raspberry Pi — одноплатный компьютер на процесорі ARM11 Broadcom BCM2835 с тактовой частотой 700 МГц и модулем оперативной памяти на 256/512МБ. Вигляд плати Raspberry Pi приведено на рис. 2.

Рис. 1. Плата Raspberry Pi

Основні характеристики даної платформи приведені в табл..1.

Таблиця 1.

Характеристики DeviceHive Discovery Platform RPi

Платформа	**DeviceHive Discovery Platform RPi**
Модель	**Raspberry Pi Model B**
Мікроконтролер	ARM11 BCM2835
Тактова частота	700 МГц
ОЗУ	256 Мбайт
Flash-память	SD карта до 16Гб
EEPROM	-
Цифрові лінії вводу / виводу	8
Канали шим	1
Інтерфейс TWI / I2C	1
Інтерфейс SPI	1
Інтерфейс UART	1
Інтерфейс USB Master	2 USB 2.0
Аналогові входи	Зовнішній АЦП і ЦАП
Відео вихід	HDMI, композитний
Аудіо вихід	HDMI, аналоговий
Мінімальне енергоспоживання	750 мА (3.75 Вт)
Напруга живлення	5 В
Порт Ethernet	10/100
Бездротові інтерфейси	Wi-Fi, Zigbee
Операційна система / Фреймворк	Raspbian
Інструменти розробки	IDLE, Scratch, Squeak/Linux

Платформа дозволяє працювати з різноманітними периферійними елементами, що розміщені на самі платі. Це дає можливість вивчати побудову систем управління об'єктом.

Перелік периферійних елементів та інтерфейси їх підключення приведені в табл. 2.

Таблиця 2.

Периферійні елементи та модулі встановлені на стенді

Датчик	Інтерфейс	Опис	Кількість
DS18B20	**1-Wire**	Цифровий термометр з програмованою роздільною здатністтю, від 9 до 12-bit, яке може зберігатися в EEPROM пам'яті приладу. DS18B20 обмінюється даними по 1-Wire шині і при цьому може бути як єдиним пристроєм на лінії так і працювати в групі.	**2**
DHT-11	**Single Wire**	Цифровий датчик температури і вологості, що дозволяє калібрувати цифровий сигнал на виході. Містить в собі АЦП для перетворення аналогових значень вологості і температури. Вимірює вологість у межах 20-90% і температуру від 0 до 50 ° С. Похибка вимірювання вологості 5%, температури 2 ° С. Час захоплення 1 сек.	**1**
DS-1307	**I2C**	Годинник реального часу з послідовним інтерфейсом, повні двійково-десятковий годинник-календар, що включають 56 байтів енергонезалежною статичної ОЗУ. Адреси і дані передаються послідовно по двухпроводній двонаправленній шині. Годинники працюють як в 24-годинному, так і в 12-годинному режимах з індикатором AM / PM. DS1307 має вбудовану схему спостереження за живленням, яка виявляє перебої живлення і автоматично перемикається на живлення від батареї.	**1**
LM-75	**I2C**	Являє собою температурний датчик, дельта-сигма АЦП і цифровий датчик перевищення робочої температури з I2C інтерфейсом. Три виводи (A0, A1, A2) доступні для вибірки за адресою. Датчик починає роботу в режимі «компаратора» при порогових значеннях за замовчуванням рівних 80 ° C TOS і 75 ° C THYST.	**1**
MCP23008	**I2C**	Інтерфейсний елемент. Розширювальний модуль вводу-виводу. Працює з інтерфейсом I2C. Призначений для збільшення числа I / O портів основного контролера. Якщо потрібні одночасно входи і виходи, а також керована підтяжка для входів, то розширювач портів це найоптимальніше рішення.	**4**
MCP23S17	**SPI**	Інтерфейсний елемент. Розширювальний модуль вводу-виводу. Працює з інтерфейсом SPI. Призначений для збільшення числа I / O портів основного контролера. Якщо потрібні одночасно входи і виходи, а також керована підтяжка для входів, то розширювач портів це найоптимальніше рішення.	**2**
Xbee	**RS232**	Модуль, який дає можливість використання протоколу ZigBee. Це стандарт бездротової передачі даних на зразок Wi-Fi і Bluetooth, але орієнтований на економію електроенергії і велику захищеність каналу при меншій швидкості. Застосована друга (Series 2), остання версія модуля. Потужності передавача вистачає для спілкування на відстані до 120 м на вулиці і до 35 м в приміщенні. Швидкість обміну даними: до 250 кбіт/с. Пристрій працює на частоті 2,4 ГГц. Можливі як прості з'єднання «точка-точка», так і мережі зі складною топологією. Модуль, споживає 45 мА в режимі прийому, 50 мА в режимі передачі і 0,01 мА в режимі енергозбереження.	**1**

Piezo	**PWM**	П'єзовипромінювач для отримання звуку.	**1**
Photosensor	**GPIO**	Фоторезистор для оцінки освітленості.	**1**
LEDs	**GPIO**	Світлодіоди. Можуть застосовуватися для індикації запрограмованих подій або при налагодженні додатків.	**13**
Buttons	**GPIO**	Клавіатура. Може застосовуватися для управління запрограмованими події або при налагодженні додатків.	**12**
Relays	**GPIO**	Часто виникає необхідність управління різними зовнішніми пристроями за допомогою включення і вимкнення напруги живлення. Причому напруга живлення і струм споживання таких пристроїв можуть змінюватися в найширших межах. Універсальним способом управління подібними пристроями є електромагнітне реле. Застосовані реле можуть комутувати навантаження в ланцюзі до 220 вольт при струмі до 5 ампер і вище.	**3**
IR receiver	**GPIO**	Спеціалізована інтегральна схема, призначена для прийому інфрачервоного сигналу від пультів дистанційного керування. На відміну від звичайного інфрачервоного фотодіода, ІЧ-приймач може приймати і обробляти інфрачервоний сигнал, що є ІЧ-імпульси фіксованої частоти і певної тривалості - пачки імпульсів. Це технологічне рішення позбавляє від випадкових спрацьовувань, які можуть бути викликані фоновим випромінюванням і перешкодами з боку інших приладів, випромінюючих в інфрачервоному діапазоні.	**1**
IR transmitter	**GPIO**	Спеціалізований модуль, призначений для відправки інфрачервоного сигналу, може виступати в якості пульта дистанційного керування. Формує інфрачервоний сигнал, що є ІЧ-імпульси фіксованої частоти і певної тривалості.	**1**

Методи вирішення задачі. Вивчення платформи складається із трьох частин:

Перша частина – вивчення периферії, програмування процедур вводу/виводу, програмування прстих процедур обробки інформації [2];

Друга частина – це побудова систем управління з використанням периферійних вузлів платформи, наприклад вимір та регулювання навколишньої температури, вологості та ін. [3];

Третя частина – це побудова більш складних систем, які обєднанні під одною назвою – інформаційно управляючі системи (інтелектуальний дім).

Нижче представлений перелік лабораторних робіт, які використовуються для програмування пристроїв зв'язку з об'єктом [3].

1. Робота з портами вводу-виводу.
2. Робота з рідкокристалічним індикатором.
3. Робота з таймером. Ініціалізація. Переривання.
4. Комплексна робота з управління об'єктом(підсумкова по трьом попереднім роботам).
5. Робота з EEPROM.
6. Робота з АЦП.
7. Робота з ЦАП.
8. Лічильник у коді Грея.
9. Двійковий лічильник.
10. Вимірювання температури.
11. Генерація звуку.
12. Комплексна робота з управління об'єктом(об'єднує попередні роботи).

Висновки та перспективи подальшого дослідження. Таким чином, одним з найважливіших стратегічних завдань на сьогоднішньому етапі модернізації післядипломної педагогічної освіти є забезпечення якості підготовки фахівців на рівні міжнародних стандартів. Надаючи школяреві можливість поглиблено вивчати цикл предметів, ми визнаємо його суб'єктивність в освітньому процесі, тобто наявність у нього власної мети, інтересів і потреб в освіті. Використання на уроках, на спецкурсах, факультативних заняттях сучасні педагогічні технології допоможуть досягнути поставленої мети у профільному навчанні.

Розв'язання завдань модернізації старшої школи можливе за умови підготовки педагогів до впровадження інноваційних педагогічних технологій.

Подальші наші дослідження стосуватимуться вивчення і визначення найбільш ефективних педагогічних технологій, шляхів та засобів опанування ними сучасними педагогами.

Даний інструментальний набір вже використовується на факультеті прикладної математики. За цей час були сформовані методичні матеріали, які допоможуть студентам в вивчені дисциплін – Периферійні пристрої, програмування пристроїв зв'язку з об'єктом, проектування вбудованих комп'ютерних систем. В подальшому ця платформа буде використання при виконанні лабораторних дисциплін Комп'ютерне забезпечення телекомунікацій.

1. DeviceHive Discovery Platform RPi [Електронний ресурс]. – Режим доступу : http://devicehive.com/documentation
2. Плахотний М. В., Наливайчук М. В., Гніденко В. В. Методичні вказівки до виконання лабораторних робіт з дисципліни «Програмування пристроїв зв'язку з об'єктом» - Київ: видавництво НТУУ КПІ, 2011. - С. 4-60.
3. *Плахотний М.В., Козьяков В.С., Наливайчук М.В., Огородницький А.Д.*ОСОБЛИВОСТІ ПОБУДОВИ ПОРТАТИВНИХ МІКРОКОНТРОЛЕРНИХ ПРИСТРОЇВ НА БАЗІ ІНТЕГРОВАНИХ ПЛАТ. //Вісник Хмельницького національного університету. – 2014. №3. –С. 53-56.

УДК УДК 004.413 (045)
Рябоконь Ю.М[1] , Жигаревич О.К.[2], Шолом П.С.[2]
[1]Національний авіаційний університет,
[2]Луцький національний технічний університет

ДОСЛІДЖЕННЯ ІСНУЮЧИХ КОМПОНЕНТІВ РОЗРОБКИ КРОССПЛАТФОРМЕРНИХ ДОДАТКІВ

Рябоконь Ю.М., Жигаревич О.К., Шолом П.С. Дослідження існуючих компонентів розробки інтерактивних кроссплатформерних додатків. У даній статті автори проаналізували результати проектування системи розробки інтерактивних додатків. Був проведений аналіз ринку інтерактивних додатків та існуючих компонентів розробки. З'ясувалось, що чим складніша система, тим більше ресурсів вона використовує.
Ключові слова: веб-застосування, база даних, сервер, клієнт, Silverlight, ASP.NET.

Рябоконь Ю.М., Жигаревич О.К., Шолом П.С. Исследование существующих компонентов разработки интерактивных кроссплатформерних приложений. В данной статье авторы проанализировали результаты проектирования системы разработки интерактивных приложений. Был проведен анализ рынка интерактивных приложений и существующих компонентов разработки. Выяснилось , что чем сложнее система , тем больше ресурсов она использует.
Ключевые слова: веб-приложения, база данных, сервер, клиент, Silverlight, ASP.NET

Ryabokon Y.M., Zhyharevych O.K., Sholom P.S. Investigatin of existing components development of interactive cross platform applications. In this article, the authors analyzed the results of system design development of interactive applications. The market of interactive applications and existing development components were analyzed. It was clarified that the more complex the system, the more resources it uses.
Keywords : web application , database, server, client , Silverlight, ASP.NET

Постановка проблеми. На сьогодні ринок інтерактивних додатків набув великих масштабів та стрімко розвивається і має широку аудиторію користувачів. Тому даний напрямок є перспективним. Для розробки ігор дається мало часу і на створення кожної гри з початку не вистачає ресурсів тоді необхідно використовувати існуючі готові рішення, на основі яких і створюється більшість інтерактивних додатків.

Готові компоненти для розробки інтерактивних додатків значно зменшують бюджет та час на виконання роботи. На даний момент існує велика кількість операційних систем, на яких виконуються ігри, і якщо писати гру окремо для кожної операційної системи це займе дуже багато часу та забере велику кількості ресурсів. Саме для уникнення таких великих витрат і використовуються компоненти для розробки інтерактивних додатків.

Гральний рушій (англ. Game engine) — програмний рушій, центральна програмна частина будь-якої відеогри, яка відповідає за всю її технічну сторону, дозволяє полегшити розробку гри за рахунок уніфікації і систематизації її внутрішньої структури. Важливим значенням рушія є можливість створення багатоплатформенних ігор (сьогодні найчастіше одночасно для ПК, PS3 та Xbox 360).

Основну функціональність гри зазвичай забезпечує рушій гри, що включає рушій візуалізатору, фізичний рушій, звук, систему скриптів, анімацію, ігровий штучний інтелект, мережевий код, керування пам'яттю, граф сцени. Часто на процесі розробки можна заощадити за рахунок повторного використання одного рушія гри для створення декількох різних ігор.

Такі системи, як Infocom'івська Z-Machine і SCI компанії Sierra, можна вважати першими закінченими гральними рушіями. Проте, термін «гральний рушій» з'явився в середині 1990-х років, головним чином, у зв'язку з 3D-іграми, такими як шутери від першої особи. Ігри Doom і Quake від id Software виявилися настільки популярними, що інші розробники замість того, щоб працювати із чистого аркуша, ліцензували основні частини програмного забезпечення й створювали свою власну графіку, персонажів, зброю й рівні — «ігровий контент» або «ігрові ресурси». Рушій Quake був використаний у більш ніж десяти проектах і дав серйозний поштовх розвиткові middleware-індустрії.

Гральні рушії також використовуються в іграх, спочатку розроблених для гральних консолей; наприклад, рушій RenderWare використовується у франчайзах Grand Theft Auto III і Burnout.

Сучасні гральні рушії — одні з найскладніших у написанні застосунків, що найчастіше складаються з десятків різних компонентів, кожний з яких можна налаштовувати окремо під потреби гри. На сайті Future Game Coders є різні теми про підсистеми сучасних ігор.

На додаток до багаторазово використовуваних програмних компонентів, ігральні рушії надають набір візуальних інструментів для розробки. Ці інструменти зазвичай складають інтегроване середовище розробки для спрощеної, швидкої розробки ігор на зразок потокового виробництва. Ці ігральні рушії іноді називають «гральним підпрограмним забезпеченням» (скор. ППЗ; англ. middleware), тому що, з погляду бізнесу, вони надають гнучку й багаторазово використовувану програмну платформу з усією необхідною функціональністю для розробки грального застосунка, скорочуючи витрати, складність і час розробки — усі критичні фактори в сильно конкуруючій індустрії відеоігор.

Як й інші ППЗ рішення, ігральні рушії зазвичай платформо-незалежні й дозволяють деякій грі запускатися на різних платформах, включаючи ігральні консолі й персональні комп'ютери, з деякими внесеними у сирцевий код змінами (або взагалі без них). Часто ігральне ППЗ має компонентну архітектуру, що дозволяє заміняти або розширювати деякі системи рушія більш спеціалізованими (і часто дорожчими) ППЗ компонентами, наприклад, Havok — для фізики, FMOD — для звуку або SpeedTree — для рендеринга. Деякі ігральні рушії, такі як RenderWare, проектують як набір слабо зв'язаних ППЗ компонентів, які можуть вибірково комбінуватися для створення власного рушія, замість традиційнішого підходу розширення або налаштовування гнучкого інтегрованого рішення. Проте розширюваність досягнута й залишається високопріоритетною в гральних рушіях через широкі можливості їхнього застосування. Незважаючи на специфічність назви, ігральні рушії часто використовуються в інших типах інтерактивних застосунків, що вимагають графіку в реальному часі, таких як рекламні демо-ролики, архітектурні візуалізації, що навчальні симулятори й середовища моделювання.

Деякі ігральні рушії надають тільки можливості 3D рендеринга в реальному часі замість усієї функціональності, необхідної іграм. Ці рушії довіряють розроблювачеві гри реалізацію іншої функціональності або її складання на основі інших гральних ППЗ компонентів. Такі типи рушіїв зазвичай відносять до «графічних рушіїв», «рушіїв рендеринга» або «3D рушіям» замість змістовнішого терміна «ігральний рушій». Однак ця термінологія використовується суперечливо: так, багато повнофункціональних гральних 3D рушіїв згадані просто як «3D рушії». Деякі приклади графічних рушіїв: RealmForge, Ogre 3D, Power Render, Crystal Space і Genesis3D. Сучасні ігральні або графічні рушії завичай надають граф сцени — об'єктно-орієнтовані представлення 3D світу гри, що часто спрощує ігровий дизайн і може використовуватися для ефективнішого рендеринга величезних віртуальних світів.

Графічний рушій (англ. graphics engine; іноді «рендерер» або «візуалізатор») — підпрограмне забезпечення (англ. middleware), програмний рушій, основним завданням якого є візуалізація (рендеринг) двовимірної або тривимірної комп'ютерної графіки. Може існувати як окремий продукт або в складі ігрового рушія. Може використовуватися для візуалізації окремих зображень або комп'ютерного відео. Графічні рушії, що використовуються в програмах, що працюють із комп'ютерною графікою (такі, як 3ds MAX, Maya, Cinema 4D, Zbrush, Blender), зазвичай називаються «рендерерами», «рисувальниками» або «візуалізаторами». Сама назва «графічний рушій» використовується, як правило, у відеоіграх.

Основна й найважливіша відмінність «ігрових» графічних рушіїв від програмних рендерерів полягає в тому, що перші повинні обов'язково працювати в режимі реального часу, тоді як другі можуть витрачати по кілька десятків годин на виведення одного зображення. Другою істотною відмінністю є те, що, починаючи приблизно з 1995—1997 років, графічні рушії роблять рендеринг за допомогою графічних процесорів (англ. GPU), які встановлені на окремих платах — відеокартах. Програмні рендерери використовують тільки центральні процесори (англ. CPU).

На етапі становлення відеоігор графічний рушій був найголовнішою частиною ігрового рушія. Властиво, приблизно 90-95 % ігрового рушія складав саме графічний рушій (іншу частину займали такі незначні підсистеми, як «система введення» і деякі інші). Однак із середини 90-х років внаслідок стрімкого розвитку відеоігор розробники ігор почали додавати у свої продукти й інші підсистеми, такі як звуковий рушій, робота з мережею. У сучасних відеоіграх графічний рушій — один із багатьох компонентів ігрового рушія (хоча й найголовніший), куди входять фізичний рушій, звуковий рушій, система анімації (кістякова й лицева анімація), система з роботи з мережею, ігровий штучний інтелект.

Як правило, графічні рушії не поширюються окремо від ігрових. Єдиного графічного рушія без додаткових компонентів й інструментарію недостатньо для створення гри, тому розроблювачі

рушіїв продають лише ігрові рушії з повним набором інструментів і компонентів. Однак це правило не відноситься до вільного програмного забезпечення. Ентузіасти створюють вільні графічні рушії й вільно їх розповсюджують. Згодом розробники ігор можуть об'єднати вільний графічний рушій із фізичним, звуковим та іншими компонентами й створити на їхній основі повноцінний ігровий рушій.

Починаючи з 2009 року, у зв'язку з розвитком графічних процесорів, а саме у зв'язку зі збільшенням їхньої багатофункціональності й гнучкості, почали розроблятися й виходити графічні рушії реального часу, які використовують потужності GPU для розрахунків. Як правило, такі рушії реалізують освітлення через метод трасування променів, а геометрія іноді представлена вокселями, а не полігонами. Дані рушії призначаються для роботи як у відеоіграх, так і в інших інтерактивних і неінтерактивних додатках, включаючи наукові розрахунки, розділення 1600*1200 пікселів, OpenGL-рендерер, 16-кратне повноекранне згладжування (FSAA), 32-бітний колір. У кадрі присутні 23653 трикутники.

На рисунку 1 зображений «Мармуровий чихуахуа» — зображення, створене з використанням вбудованого рендерера Blender.

Рис.1. Графічний мармуровий чихуахуа

OptiX — графічний рушій реального часу, розроблений nVidia, використовує CUDA, і працює винятково на графічних процесорах виробництва nVidia і призначений для різноманітних обчислень, досліджень і моделювань. «OptiX» є гібридним рендерером — основним є використання трасування променів, але є присутнім і растеризація.

Octane Render — графічний рушій реального часу, розроблений компанією Refractive Software LTD, що використовує CUDA і працює на всіх графічних процесорах nVidia, починаючи з 8X00. Використовує трасування променів.

id Tech 6 — графічний рушій, що входить до складу ігрового рушія id Tech 6, буде використовувати трасування променів і вокселі.

Фізичний рушій (англ. physics engine) — програмний рушій, що робить симуляцію фізичних законів реального світу у світі віртуальному з тим або іншим ступенем апроксимації. Найчастіше фізичні рушії використовуються не як окремі самостійні програмні продукти, а як складені компоненти (підпрограми) інших програм. Усі фізичні рушії умовно діляться на два типи: ігрові й наукові. Перший тип використовується в комп'ютерних іграх як компонент ігрового рушія. У цьому випадку він повинен працювати в режимі реального часу, тобто відтворювати фізичні процеси в грі з тою ж швидкістю, з якою вони відбуваються в реальному світі. Разом із тим, від ігрового фізичного рушія не потрібно точності обчислень. Головна вимога — візуальна реалістичність, — і для її досягнення не обов'язково проводити точну симуляцію. Тому в іграх використовуються дуже приблизні апроксимації, наближені моделі й інші програмні «трюки». Наукові фізичні рушії використовуються в науково-дослідних розрахунках і симуляціях, де вкрай важлива саме фізична точність обчислень. Разом із тим швидкість обчислень не грає істотної ролі.

Сучасні фізичні рушії симулюють не всі фізичні закони реального світу, а лише деякі, причому із часом і прогресом у галузі інформаційних технологій і обчислювальної техніки список «підтримуваних» законів збільшується. На початок 2010 року фізичні рушії можуть симулювати такі фізичні явища й стани:

- динаміка абсолютно твердого тіла;
- динаміка деформованого тіла;
- динаміка рідин;
- динаміка газів;
- поведінка тканин;
- поведінка мотузок (тросів, канатів тощо).

У серпні 2009 року англомовний журнал Game Developer (англ.), присвячений розробці комп'ютерних ігор, опублікував статтю про сучасні ігрові рушії та їхнє використання. Згідно з даними журналу, найпопулярнішим серед розробників є рушій nVidia PhysX, що займає 26,8% ринку. На другому місці перебуває Havok, що займає 22,7% ринку. Третє місце належить рушію Bullet Physics Library (10,3%), а четверте — Open Dynamics Engine (4,1%).

Фізичний рушій дозволяє створити деякий віртуальний простір, який можна наповнити тілами (віртуальними статичними й динамічними об'єктами), і вказати для нього якісь загальні закони взаємодії тіл і середовища, тією, чи іншою мірою наближені до фізичних, задаючи при цьому характер і ступінь взаємодій (імпульси, сили тощо). Властиво розрахунок взаємодії тіл рушій і бере на себе. Коли простого набору об'єктів, що взаємодіють за певними законами у віртуальному просторі, недостатньо в силу неповного наближення фізичної моделі до реального світу, можливо додавати (до тіл) зв'язки. Розраховуючи взаємодію тіл між собою й із середовищем, фізичний рушій наближає фізичну модель одержуваної системи до реального світу, передаючи уточнені геометричні дані засобові відображення (рендереру).

Ігровий штучний інтелект (англ. Game artificial intelligence) — набір програмних методик, які використовуються у відеоіграх для створення ілюзії інтелекту в поведінці персонажів, керованих комп'ютером. Ігровий ШІ, крім методів традиційного штучного інтелекту, включає також алгоритми теорії керування, робототехніки, комп'ютерної графіки та інформатики у цілому.

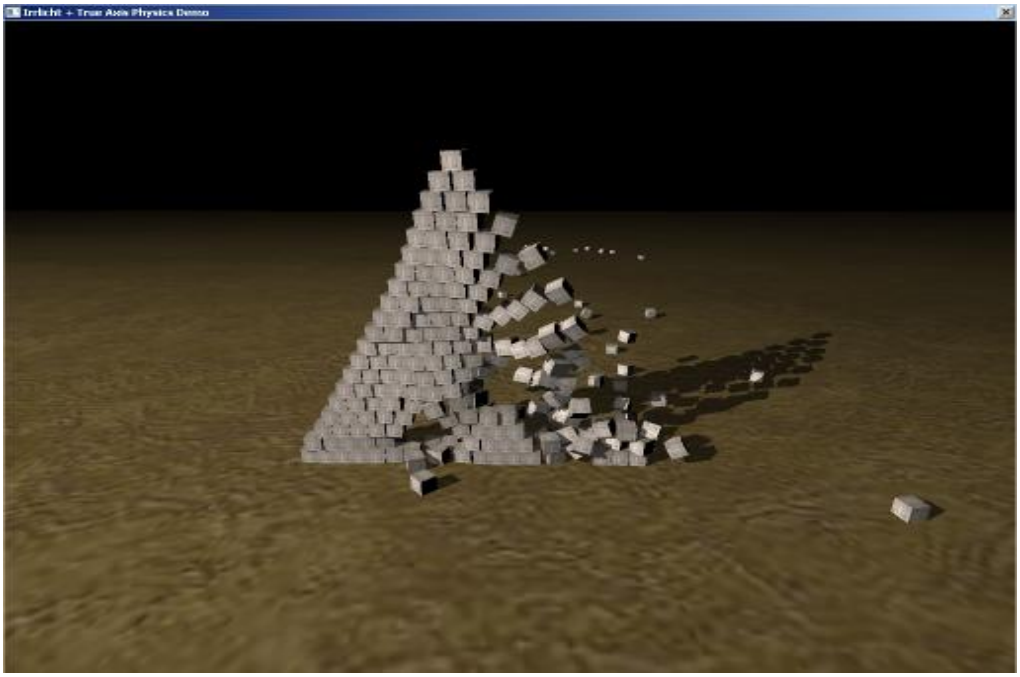


Рис. 2. Приклад роботи фізичного рушія

Реалізація ШІ сильно впливає на геймплей, системні вимоги і бюджет гри, і розробники балансують між цими вимогами, намагаючись зробити цікавий і невимогливий до ресурсів ШІ малою ціною. Тому підхід до ігрового ШІ серйозно відрізняється від підходу до традиційного ШІ — широко застосовуються різного роду спрощення, обмани й емуляції. Наприклад: з одного боку, в шутерах від першої особи безпомилковий рух і миттєве прицілювання, властиве ботам, не

залишає жодного шансу людині, так що ці властивості штучно знижуються. З іншого боку — боти повинні робити засідки, діяти командою й т.д., для цього застосовуються «костилі» у вигляді контрольних точок, розставлених на рівні.

Деякі ігрові програмісти розглядають будь-яку методику, що використовується для створення ілюзії інтелекту, як частину ігрового ШІ. Однак цей погляд є спірним, тому що він включає методики, які широко використовуються поза рушієм ігрового ШІ. Наприклад, інформація про потенційні майбутні зіткнення є важливою інформацією, що вводиться в алгоритми, які допомагають створювати ботів, що будуть досить розумними для уникнення зіткнень з об'єктами. Але ті ж самі методики виявлення зіткнень є необхідним і одним із найважливіших компонентів фізичного рушія. Точно так само результати іспитового напрямку погляду бота звичайно є важливими даними, що вводяться в систему прицілювання бота; разом із тим ці дані широко використовуються при рендерингу в графічному рушії. Фінальним прикладом є скриптинг, що може бути зручним інструментом для всіх аспектів ігрової розробки, однак часто сильно асоціюється з контролюванням поведінки неігрових персонажів.

Пуристи вважають, що вираз «штучний інтелект» у терміні «ігровий штучний інтелект» є перебільшенням, оскільки ігровий ШІ описує не інтелект і використовує тільки деякі з напрямків академічної науки «Штучний інтелект». Беручи до уваги, що «реальний» ШІ звертається до галузей систем, що самонавчаються, і прийняття рішень, які базуються на довільному уведенні даних, і навіть до остаточної мети «сильного» ШІ, що може мислити, ігровий ШІ часто складається з декількох емпіричних правил і евристики, яких досить, щоб надати гравцеві гарний геймплей, відчуття й враження від гри.

Покращення розуміння академічного ШІ розробниками ігор і зростаючий інтерес академічного співтовариства до комп'ютерних ігор викликає питання, наскільки й у якій мірі ігровий ШІ відрізняється від класичного. Однак, істотні розходження між різними прикладними галузями штучного інтелекту означають, що ігровий ШІ все ще може бути розглянутий як окрема підгалузь ШІ. Зокрема, здатність «законним» чином вирішити деякі проблеми ШІ в іграх через обман утворює важливе розходження. Наприклад, виведення позиції невидимого об'єкта з минулих спостережень може бути важкою проблемою, коли ШІ застосований до робототехніки, але в комп'ютерних іграх неігровий персонаж може просто шукати позицію в ігровому графі. Такий обман може призвести до нереалістичної поведінки й тому не завжди бажаний. Але його здатність служити для розрізнення ігрового ШІ веде до нових проблем, таких як: коли і як використовувати обман.

Евристичні алгоритми ігрового штучного інтелекту використовуються в широкій розмаїтості в багатьох галузях усередині гри. Найочевидніше застосування ігрового ШІ проявляється в контролюванні неігрових персонажів, хоча скриптинг теж є дуже розповсюдженим способом контролю. Пошук шляху є іншим широко розповсюдженим застосуванням ігрового ШІ, — він особливо проявляється в стратегіях реального часу. Пошук шляху є методом для визначення того, як неігровому персонажеві перейти з однієї точки на мапі до іншої: потрібно враховувати ландшафт, перешкоди й, можливо, «туман війни». Ігровий ШІ також пов'язаний із динамічним ігровим балансуванням.

Концепція непередбачуваного (англ. emergent) ШІ була недавно досліджена в таких іграх як Creatures, Black & White і Nintendogs і в таких іграшках, як тамагочі. «Свійські тварини» у цих іграх мають здатність «навчатися» на діях, вчинених гравцем, і їхня поведінка змінюється відповідно. У той час, як ці рішення взяті з обмеженої множини можливих рішень, це дійсно часто дає бажану ілюзію інтелекту по іншу сторону екрана.

При розробці ігор звуковий супровід завжди перебувало кілька на другому плані. Розробники вважають за краще витратити час на введення новомодних ефектів для 3D -графіки , тоді як реалізація звуку пускається на самоплив. Людей складно переконати витратити час і кошти на якісний звук в грі. Разом з тим, більшість користувачів також більш охоче витратять гроші на новітній 3D-акселератор, ніж на нову звукову карту.

Однак, ситуація змінюється: останнім часом звуку стало приділятися набагато більше уваги і з боку користувачів, і з боку розробників. У сучасних проектах звуку відводиться до 40 відсотків бюджету і тимчасово - людських ресурсів.

Виробники звукових чипів і розробники технологій 3D - звуку також доклали чимало зусиль , щоб переконати користувачів і розробників додатків в тому, що хороший 3D - звук є невід'ємною частиною сучасного комп'ютера.

Звук з стерео перетворився на тривимірний, потім з'явилися і багатоканальні рішення: 4 - канальні, 5.1 - звук, а останнім часом і 7.1.

Саме поняття «тривимірний звук» має на увазі, що джерела звуку розташовуються в тривимірному просторі навколо слухача. При цьому кожен джерело являє собою в широкому сенсі будь-який об'єкт у віртуальному ігровому світі, здатний виробляти звуки.

Джерела звуку (Sound Sources). Причому, деякі з джерел звуку - звичайні стерео, такі як фонова музика (в даній конкретній грі це main ambient: вітер або трек «звуки джунглів»), 8 джерел створюються ігровими персонажами - монстрами, 1 джерело створює безпосередньо гравець - постріл, звуки кроків і 3 випадкових джерела - для посилення ефекту простору (ambient sounds) - в даному випадку «джунглі»: звуки комах, птахів і т.п.

3D - звук потрібен для більш глибокого занурення користувача в віртуальний світ гри, за рахунок посилення реалізму відбувається на ігровій сцені. Для цього використовуються різні технології, адаптовані або гіперболізує поведінку звуку в реальному світі. Наприклад, реверберації, відбиті звуки, оклюзії (звук, що пройшов через перешкоду), обструкції (звук не пройшов через перешкоду), дистанційне моделювання (вводиться параметр віддаленості джерела звуку від слухача) і маса інших ефектів.

Звичайно ж, сприйняття звуку строго індивідуально у кожної людини (воно залежить від форми вуха, віку та психологічного настрою в конкретний момент). Таким чином, ніколи не буде однозначної думки про звучання тієї чи іншої звукової карти або ефективності тієї чи іншої технології 3D-звуку. Відтворення ж звуку сильно залежить від звукової карти і колонок, а також від конкретної реалізації звукового движка в грі.

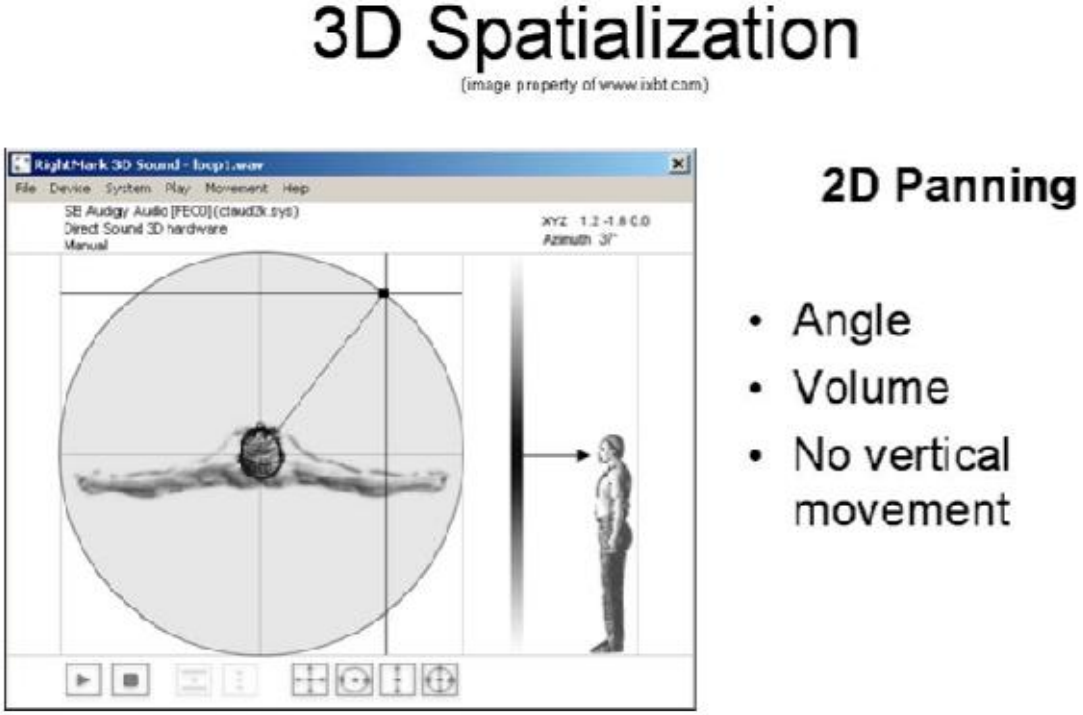


Рис.3. Представлення сприйняття звуку людиною

Розглянемо, як же створюється ефект 3D - звуку. На рисунку 3 представлено сприйняття звуку людиною. Почнемо з найпростішої технології 2D - панорамування. Ця технологія застосовувалася ще в DOOM від відомих id Software - все просто: кожне джерело звуку грається як стерео, а позиціонування створюється за рахунок зміни гучності лівого або правого каналів. У такій системі немає вертикального позиціонування, але можлива реалізація ефектів, пов'язаних, скажімо, з невеликою зміною (фільтрацією високих частот) звуку, коли він знаходиться ззаду слухача, так як в реальності ми чуємо злегка заглушений звук, якщо він знаходиться за головою слухача.

Тепер ми підійшли до апаратної реалізації. На двох колонках або в навушниках звукова карта відтворює положення джерела звуку за допомогою HRTF (Head Related Transfer Function). Фільтрація та інші перетворення відтворюють поведінку слухової системи людини.

На рисунку 3 зображені три HRTF (позиція джерела звуку азимут 135 градусів і 36 градусів) трьох різних людей для лівого і правого вуха відповідно. Можна помітити певні закономірності на всіх трьох функціях. Звідки ж вони беруться? У більшості своїй вони записуються за допомогою спеціальних методик і спеціальних стереомікрофоном, які вставляються у вуха людини або спеціального манекена (KEMAR). Фірма Sensaura, наприклад, використовує синтетичні HRTF, використовуючи ті самі закономірності, які ми можемо спостерігати на ілюстрації. Наприклад, пік в районі 2500 Гц і спад близько 5000 Гц для даної точки в просторі. Інші фірми використовують усереднені HRTF .

По суті, вся система - це два FIR - фільтра (Finite Impulse Response), передавальна характеристика яких і є HRTF. Так як HRTF дискретні, і зберігати мегабайти HRTF для часток градуса важко - реальне положення джерела прораховується інтерполяцією HRTF.

OGRE (англ. Object-Oriented Graphics Rendering Engine, об'єктно орієнтований графічний рушій) — гнучкий, орієнтований на сцену та кросплатформений графічний рушій (на відміну від рушія гри) написаний на C++ та спроектований так, щоб зробити простішим та інтуїтивним процес розробки програм, що використовують тривимірну графіку. Поширюється на правах MIT ліцензії. Бібліотека класів спроектована таким чином, що її можна однаково використовувати з OpenGL та Direct3D не змінюючи програмного коду прикладної програми.

Серед комерційних ігор, які використовують OGRE можна відмітити: Ankh, Torchlight та Garshasp.

OGRE сам по собі не є ігровим рушієм і за заявою автора ніколи таким не буде. OGRE був, є і буде графічним рушієм для рендеринга тривимірної графіки. Велику популярність рушій отримав за рахунок своєї гнучкості, що дозволяє «схрещувати» його з багатьма іншими бібліотеками (фізика — ODE, Newton, PhysX, Bullet; звук, мережа, графічний інтерфейс тощо). На рисунку 2.7 представлений приклад роботи OGRE.

Для реалізації графічного інтерфейсу користувача (англ. GUI — Graphic User Interface) можуть застосовуватися як стандартні (недостатньо добре реалізовані, і, за словами розробників, в майбутньому, можливо, будуть виключені) функції графічного інтерфейсу OGRE, так і імпортуватися сторонні бібліотеки (OpenGUI, MyGUI (дуже популярний серед учасників російського співтовариства OGRE), CEGUI).

OGRE є вільним програмним забезпеченням, поширюваним під ліцензіями LGPL/MIT і має дуже активне співтовариство.

Можливості:

- підтримка платформ Windows, Linux та Mac OS X;
- скриптова система управління матеріалами (мультитекстурування, мультипрохідне змішування);
- завантаження текстур у форматі PNG, JPEG, TGA, BMP або DDS, підтримка стислих текстур (DXT/S3TC);
- експортери для основних комерційних та вільних пакетів 3D моделювання;
- система управління ресурсами;
- підтримка DirectX, OpenGL;
- підтримка шейдерів, написаних на асемблері або мовах високого рівня: Cg, DirectX HLSL або GLSL;
- складна скелетна анімація (анімація тіла), анімація гнучких форм, морфінг (анімація особи), анімація шляху (камера, переміщення).

Irrlicht (Irrlicht Engine) - тривимірний графічний движок, який є безкоштовним вільним програмним продуктом і поширюється на умовах ліцензії zlib.

Irrlicht використовує можливості OpenGL, DirectX і декількох власних рендерерів. Користувачеві надаються різні функціональні можливості по завантаженню та управління тривимірними (3D) об'єктами (сцени, моделі тощо), небагатьма спецефектами і графічним інтерфейсом користувача. Рекомендується для ознайомлення з процесом розробки і створення нескладних ігор і демосцен (Irrlicht підтримує формати популярних ігор і движків , зокрема моделі quake 2, quake 3, карти рівнів та ін). Не вимагає підключення сторонніх модулів для

реалізації високорівневих функцій (є найпростіша фізика, GUI (графічний інтерфейс користувача) і т. д.). Існує три офіційні доповнення для Irrlicht: IrrKlang (аудіобібліотеку), IrrXML (завантаження і обробка XML - файлів), IrrEdit (редактор сцен). Для використання розширених функцій фізики існує фізичний движок ChronoEngine (з причини того, що в Irrlicht вбудована примітивна фізична система).

Одна з важливих особливостей Irrlicht його кроссплатформенність - тобто здатність працювати на різних платформах. Платформо - незалежний прошарок забезпечує легке перенесення на різні, які не підтримує офіційно платформи, зокрема існують порти під android, iPhone і інші.

Marmalade SDK - кроссплатформне SDK від Ideaworks3D Limited. Являє собою набір бібліотек, зразків, інструментів і документацій необхідних для розробки , тестування та розгортання додатків для мобільних пристроїв. Основоположною концепцією Marmalade SDK є одноразове написання програми і компілювання її на всі підтримувані платформи, без необхідності програмування на різних мовах програмування і використання різних API для кожної платформи.

Для того щоб використовувати Marmalade SDK необхідно придбати ліцензію. Є чотири види ліцензій, які надають доступ до різних наборів платформ, розгортання та рівня технічної підтримки. Ліцензія потрібна для кожного комп'ютера де встановлений Marmalade SDK.

Основа Marmalade SDK складається з двох основних шарів. Низькорівневий з API називається Marmalade System забезпечує рівень абстракцій, що дозволяє отримати програмісту доступ до функціональності пристрою, таких як управління пам'яттю, доступ до файлів, мережі, даними введення (таким як , акселерометр, клавіатура, сенсорний екран), звуку.
Marmalade Studio C ++ API , який забезпечує високорівневу функціональність, в основному спрямований на підтримку 2D (наприклад, обробка растрового зображення і шрифтів) і 3D-рендрингу графіки.

Nebula Device - ігровий движок, розроблений німецькою компанією Radon Labs і вперше використаний в комп'ютерній грі 2002 Project Nomads. Nebula Device вільним програмним продуктом і поширюється на умови ліцензії MIT. При розробці Nebula Device основний напрямок робився на оптимальну роботу з великими відкритими просторами, ефекти візуалізації неба і велику дистанцію промальовування. Зараз Nebula Device не тільки використовується в усіх іграх компанії Radon Labs, але і в численних сторонніх розробках.

Движок написаний на мові програмування C++ і підтримує кілька скриптових мов, таких як Tcl, Lua, Python, Ruby, Java і . NET Framework. Є можливість підключити і другі скриптові мови, за допомогою під'єднуваних плагінів. Рендеринг движка функціонує у двох режимах (DirectX і OpenGL), завдяки чому забезпечується кроссплатформенність .

Підтримуються операційні системи Linux, Mac OSX, IRIX і Microsoft Windows а також ігрова приставка Xbox. Для текстур підтримуються графічні формати DDS, BMP, JPEG, GIF, TIFF, PNG і деякі інші. Відкритість графічних форматів дає деяку творчу свободу ентузіастам , які розробляють модифікацій для ігор на движку Nebula Device.

Підтримувані формати тривимірних моделей - NVX, N3D і OBJ. Nebula Device дозволяє також використовувати шейдерні ефекти, скелетну анімацію, системи частинок, динамічні тіні та пост- ефекти. До складу SDK входять також додаткові утиліти, такі як програма для контролювання джерел світла «Light Control Tool»; архів з вихідним кодом також можна завантажити окремо. До вільної завантаження доступні три покоління ігрового движка.

Висновки. У статті було здійснено дослідження системи компонентів розробки інтерактивних додатків, яка включає в себе велику кількість зв'язаних модулів. Модулі графічного відображення, фізичного моделювання, моделювання звуку, штучний інтелект.

Була розглянута достатня кількість існуючих альтернативних компонентів, кожний з яких має свої переваги та недоліки. Система, що розробляється буде брати основні принципи архітектури з даних компонентів та додавати свої рішення, які будуть сприяти зменшенню входження для використання компонентів.

1. Бабенко Л.П., Лавріщева К.М. Основи програмної інженерії: Навч. посіб. – К.:Т-во «Знання», 2001. – 269 с.

2. М. Мак-Дональд, М.Шпушта Microsoft ASP.NET 3.5 с примерами на C# 2008 и Silverlight 2 для профессионалов = Pro ASP.NET 3.5 in C# 2008: Includes Silverlight 2. — 3-е издание. — М.: «Вильямс», 2009. — С. 1408.
3. Мэтью Мак-Дональд. Silverlight 3 с примерами на C# для профессионалов = Pro Silverlight 3 in C#. — 3-е изд. — М.: Вильямс, 2010. — 656 с.
4. Мэтью Мак-Дональд Microsoft ASP.NET 2.0 с примерами на C# 2005 для профессионалов.: Пер. с англ. – М.:ООО «И.Д. Вильямс», 2006. –1408с.
5. Роб Камерон, Дэйл Михалк ASP.NET 3.5, компоненты AJAX и серверные элементы управления для профессионалов = Pro ASP.NET 3.5 Server Controls with AJAX Components. — М.: «Вильямс», 2009. — С. 608.
6. Дейт К. Дж. Введение в системы баз данных. — 8-е изд. — М.: Вильямс, 2006. — 1328 с.

УДК 517.977
Семенюк В.Я., Шолом П.С., Машевський М.В.
Луцький національний технічний університет, Луцьк

ОПТИМІЗАЦІЯ ЩОДЕННОГО ПЛАНУ ВИПУСКУ ПРОДУКЦІЇ МЕТОДОМ ГІЛОК І МЕЖ

Семенюк В.Я., Шолом П.С., Машевський М.В. Оптимізація щоденного плану випуску продукції методом гілок і меж. На основі методу гілок і меж побудовано систему відшукання оптимального плану виробництва продукції, яка мінімізує час на переналаштування системи під виготовлення продукції. Розглянуто 10 видів виробів і визначено час на переналаштування верстату на кожен з них. Розроблено алгоритм та реалізовано обчислювальну процедуру щодо розрахунку оптимального плану виробництва. Проведено оцінювання витрат часу на переналаштування виробничих ліній на інші види товарів.
Ключові слова: метод гілок і меж, оптимізація виробничої системи, алгоритм знаходження мінімальних і максимальних оцінок часу.

Семенюк В.Я., Шолом П.С., Машевский М.В. Оптимизация суточного плану производства продукции методом вервей и границ. Используя метод ветвей построено систему поиска оптимизированого плана производства продукции, которая минимизирует время на пуско-настроечные работы. Рассмотрено 10 видов изделий и определено время на перенастройку станка на каждый из них. Разработан алгоритм и вычислительная процедура поиска оптимального плана производства. Сделана оценка использования времени на смену оборудования под производство других товаров.
Ключевые слова: метод ветвей, оптимизация производственной системы, алгоритм поиску минимальных и максимальных оценок времени.

Semenyuk V.Y., Sholom P.S., Mashevskij M.V. Daily plan Optimization of the production by the branch and bound method. Using the method of branches and bounds a system of finding the optimal plan of the production, that minimized the time for the starting-adjusting works, is designed. 10 types of products is considered and the time to reconfigure the machine to each of them is defined. The algorithm and the computational procedure for finding the optimal production plan is designed. The estimation of the use of time to replace equipment for the production of the other goods is done.
Keywords: method of branches, optimization of the production system, the minimum and maximum time estimates finding algorithm.

Постановка наукової проблеми. Розглянуто виробничу систему, при складанні плану роботи якої необхідно провести оптимізацію щоденного плану випуску продукції так, щоб сума часу простою виробництва була мінімальною. Це дозволить скоротити кількість універсальних верстатів, необхідних для виробництва, а також оптимізувати роботу налагоджувальної служби.

Час переналаштування

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
A1	###	15	15	45	20	20	35	25	20	25
A2	25	###	15	30	15	40	15	25	45	25
A3	40	25	###	35	40	40	25	20	25	30
A4	20	40	20	###	30	20	45	25	40	45
A5	30	45	40	15	###	15	20	30	15	35
A6	15	40	35	40	35	###	30	20	35	35
A7	45	35	45	20	35	25	###	15	40	30
A8	45	30	35	45	35	25	15	###	35	45
A9	40	40	20	20	45	40	35	45	###	35
A10	25	15	35	20	45	20	20	30	45	###

Рис. 1. Час переналаштувань системи.

На графічному представленні (рис. 1) показано можливість переналаштування верстату з випуску продукції A_i на випуск A_j. За час $a(A_i, A_j)$ приймем час переналаштування верстату з випуску продукції A_i на випуск A_j. Матриця відстаней між вершинами не буде симетричною. При розгляді самого простого випадку, коли для випуску деякого виду продукту потрібно встановити на верстат

деяке додаткове обладнання, а також налаштувати його, а для переходу на виготовлення першого продукту необхідно лиш демонтувати встановлену деталь, можна пересвідчитись у хибності думки про симетричність системи.

Окрім часу є ще кількість одиниць продукції кожного виду, яку необхідно виготовити за зміну, і час на виготовлення одиниці продукції. Не втрачаючи загальності в алгоритмі будемо рахувати відразу час, необхідний на виготовлення усієї партії товару даного виду. При ідеальних умовах, коли ми маємо n видів товарів і на виготовлення кожного виду товару тратиться якраз одна зміна, то ніяких затрат часу на переналаштування верстатів не буде. В сучасних умовах рідко де зустрічаються такі виробничі системи. Як правило універсальні верстати потрібно кілька разів переналаштовувати під виробництво наступного виду продукції. Виготовляти деяку партію товару частинами відразу на кількох верстатах, при можливості виготовити її на одному за зміну теж недоцільно.

Виникає задача мінімізувати час, який витрачається на переналаштування, щоб задіяти мінімальну кількість працівників на виробництві та обійтись якомога меншою кількістю обладнання. Такі задачі часто можна зустріти в сучасних умовах Волині, наприклад для пакування на ТОВ «ПАККО Холдинг» та виготовлення кабелю на ТОВ «Кромберг енд Шуберт Україна».

Так як в якості критерію оптимальності в задачі виготовлення продукції прийнятий мінімум сумарного часу переналаштувань верстатів, задача зводиться до наступної: знайти мінімум цільової функції

$$\sum_{k=1}^{n} a_{i_k j_k}. \qquad (1)$$

Предметом даного дослідження є метод віток і меж знаходження такого плану виробництва, що забезпечить мінімальний час переналаштувань.

Аналіз останніх досліджень. Деякі з NP-повних задач для вирішення використовують метод віток і меж. У більшості задач основним способом вирішення є повний перебір. Для зменшення часу знаходження оптимуму використовують метод віток і меж. Дана задача є однією з підзадач планування гнучких виробничих ліній. Формалізованих алгоритмів знаходження оптимальних планів даної задачі не існує. В більшості випадків ці задачі розв'язуються вручну і найчастіше методом повного перебору. В даній статті ми спробуємо дати опис алгоритму розв'язання класу однотипних задач де є універсальні верстати з можливістю переналаштування і час переналаштувань верстата з будь-якого стану в інші . Розроблене програмне забезпечення дозволить знаходити план випуску продукції на день з мінімальним часом простою.

Основні допущення та рівняння. Алгоритм повинен дати відповідь на питання знаходження мінімуму функції (1) на допустимій множині розв'язків системи. Операції розбиття вихідної множини на підмножини(гілки), та знаходження оцінок(меж) є основою алгоритму. Існує оцінка множини згори та оцінка знизу. Оцінка згори – точка, що гарантовано не менша за максимум на заданій підмножині. Оцінка знизу – точка, що гарантовано не більша за максимум на заданій підмножині.

Припустимо, що у нас є однакова кількість верстатів і типів продукції, тоді для виготовлення кожного з видів товару ніяких переналаштувань не потрібно. Або система замовлень є незмінною і знайдений оптимальний план використовується з дня в день. В таких випадках ніяке програмне забезпечення не потрібне. Проте в більшості випадків системи вимагають щоденних коригувань у планах випуску продукції, що зумовлено постійним оновленням списку продукції та зміною попиту на деякі з видів продукції. Оскільки щоденний план випуску продукції з ростом інформаційних технологій можна коригувати до одиниць, то програмне забезпечення, що дозволяє мінімізувати простої виробничих ліній є досить актуальним.

Першим підходом до вирішення цієї задачі було саме ручне її вирішення з використанням повного перебору. Було розроблено програмне забезпечення, яке допускало ввід часу переналаштувань системи і часу на виготовлення партії кожного з видів товару. На виході ми отримували оптимальний план роботи на день. При знаходженні оптимуму було використано повний перебір. Спочатку оптимум знаходився для першого верстату, використовуючи перебір множини усіх видів продукції. При виборі продукції для виготовлення на першому верстаті виникала та ж сама задача, тільки з меншою кількістю видів продукції. При виборі деякого плану крім сумарного часу виготовлення продукції, якщо він був менший за робочу зміну, брався ще і час на переналаштування системи. Цей час рахувався уже з використанням методу гілок і меж.

Отже, для кожного вибору видів продукції ми мали сумарний час затрачений на виготовлення та мінімальний час на переналаштування системи. Оптимум шукався за двома параметрами: час роботи, який затрачений на виготовлення і переналаштування, менший за робочу зміну у 8 годин, при цьому час затрачений власне на виготовлення продукції є максимальний.

Видно, що перебір одного і того ж продукту використовувався досить багато разів. Окрім того при оптимальному плані випуску для першого верстату з кожним наступним верстатом загальний план погіршувався, і могли виникати досить великі простої, пов'язані з переналаштуванням системи. Тому не можна було стверджувати, що знайдений план оптимальний.

Цікавим рішенням стала зміна етапів. Спочатку був знайдений мінімальний час затрачений на переналаштування системи, а далі вже заповнювались зміни для верстатів.

Мінімізація часу затраченого на переналаштування системи розв'язується як класична задача комівояжера. Так як усі відомі методи відшукання мінімального шляху не дають перебору усіх вершин графа, а розв'язком задачі комівояжера є повний зв'язний цикл.

Задачу комівояжера формулюють так: дано множину міст, а також відстань між усіма можливими парами міст. Необхідно знайти шлях, який пролягає через усі міста, та повертається у початкове, окрім того сумарна довжина пройденого шляху має бути мінімальною.

Задачу можна представити у вигляді моделі на графі. Розрізняють різні варіанти задачі, найважливішими з яких є симетрична та асиметрична задачі. У випадку симетричної задачі всі пари ребер між тими самими вершинами мають однакову вагу. В симетричному випадку кількість можливих маршрутів вдвічі менша. [1] В нашому випадку як було зазначено вище задача асиметрична.

Всі існуючі методи розв'язування задачі комівояжера можна поділити на дві групи: методи знаходження оптимальних маршрутів; наближені методи [5].

Серед алгоритмів пошуку оптимального розв'язку задачі комівояжера відомими є метод гілок та меж, прогресивні методи покращення на основі методів лінійного програмування, комбінації методів гілок та меж та відсікальних площин. Але всі їх поєднує той факт, що вони не застосовні до задач великої розмірності, оскільки обчислювальна складність цих алгоритмів зростає експоненційно. Вважаючи що наша система не така вже і велика можемо легко обійтись і ними, а саме використаний був метод гілок і меж.

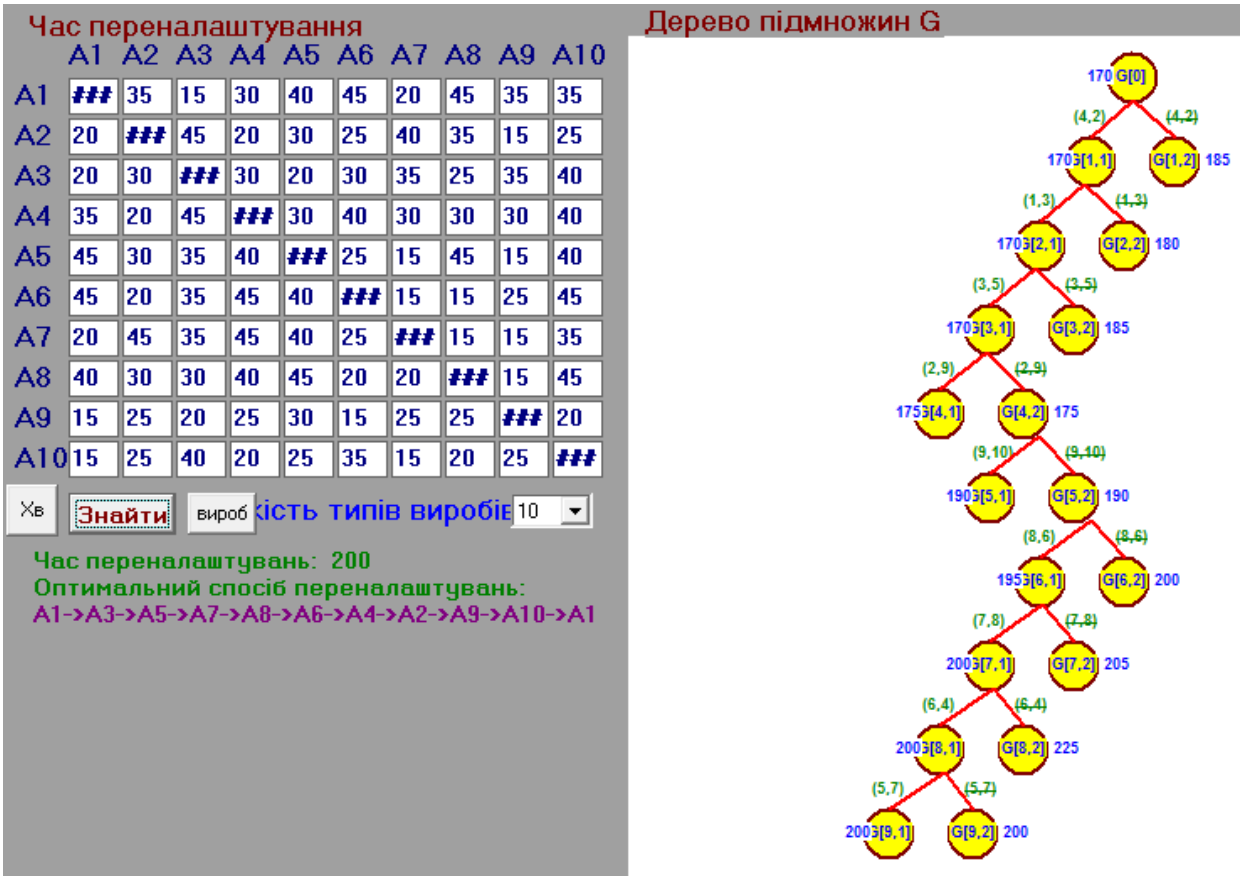
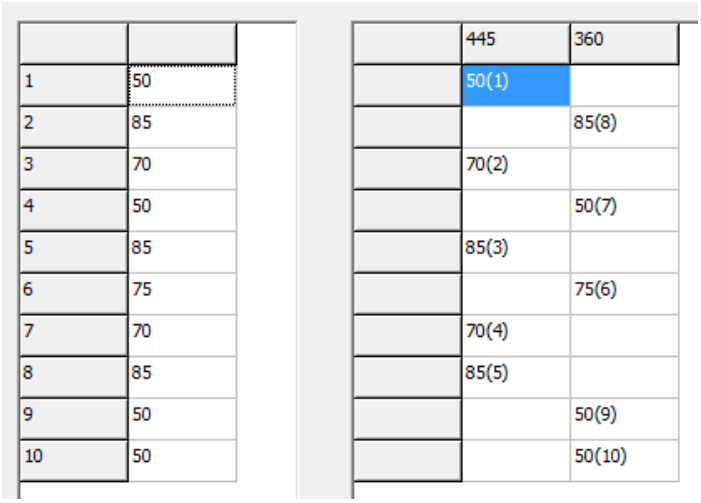


Рис. 2. Гілкування

На рисунку видно, що для заданої системи було знайдено оптимальний план переналаштувань і час затрачений на це рівний 200хв. Крім оптимального плану було знайдено і варіанти гілкувань.

```
for (i=1, N, ++) {
   for (j=1, N, ++){
    PK[i,j]=-1;}}
  for (i=1, N, ++) {
   for (j=1, N, ++){
    if GIJ[i,j]==0 {
     xmin=9999; ymin=9999;
     for (i=1, N, ++) {
      if (GIJ[i,l]<=xmin) and (GIJ[i,l]<>-1) and (l<>j) {
       xmin=GIJ[i,l];}
      if (GIJ[l,j]<=ymin) and (GIJ[l,j]<>-1) and (l<>i) {
       ymin=GIJ[l,j];}
          if xmin==9999 { xmin:=0;}
     if ymin==9999 { ymin:=0;}
     PK [i,j] =xmin+ymin;}
    max:=-1;
for (i=1, N, ++) {
   for (j=1, N, ++){
    if PK [i,j]>max {
     max:=PK [i,j];
     r:=i; m:=j;
    }}
```

Розроблене програмне забезпечення дало відповідь на план випуску продукції (рис. 3).



1	50
2	85
3	70
4	50
5	85
6	75
7	70
8	85
9	50
10	50

	445	360
	50(1)	
		85(8)
	70(2)	
		50(7)
	85(3)	
		75(6)
	70(4)	
	85(5)	
		50(9)
		50(10)

Рис. 3. Оптимальний план та час завантаження кожного з верстатів

В лівій частині рисунку відображено законтрактований план випуску продукції на день. Для здійснення цього плану необхідно 2 верстати. Час роботи кожного з них відображено зверху. Нагадаємо, що час роботи відображається у хвилинах і не повинен перевищувати одну зміну, тобто 480 хвилин. В дужках відображено черговість зміни виробництва продукції.

```
form3.StringGrid2.Cells[j,0]:= floattostr(s);
    s:=s+ strtoint(form3.StringGrid1.Cells[1,NewPut[i]])+
    strtoint (ObjEdit('Edit',NewPut[i],NewPut[i+1]).Text) ;
    if s<max then    begin
form3.StringGrid2.Cells[j,NewPut[i]]:=form3.StringGrid1.Cells[1,NewPut[i]]+'('+ inttostr(i)+')';
     i:=i+1;  end
```

```
    else begin j:=j+1; form3.StringGrid2.ColCount:=form3.StringGrid2.ColCount+1;
    s:=strtoint(form3.StringGrid1.Cells[1,NewPut[i]])+
    strtoint (ObjEdit('Edit',NewPut[i],NewPut[i+1]).Text) ;
form3.StringGrid2.Cells[j,NewPut[i]]:=form3.StringGrid1.Cells[1,NewPut[i]]+'('+ inttostr(i)+')';
     i:=i+1;end;
 end;
```

Єдине, що не враховує дана програма – це час простою, коли майстер зайнятий на іншому верстаті. Якщо продукція що випускається не одноразовий випуск, то простоїв завжди можна уникнути збільшивши кількість виготовленої продукції деяких з видів товарів.

Висновки та перспективи подальших досліджень. На основі підходів теорії дослідження операцій побудовано алгоритм відшукання оптимального плану виробництва досить широкого класу задач та побудовано програмну модель розрахунків. Дана модель враховує зменшення кількості розрахунків, максимальні і мінімальні оцінки функціонування системи, які встановлюються конкретними технологічними показниками.

Проведено оцінювання оптимального плану виробництва продукції для конкретного технологічного процесу.

Відповідно до запропонованої математичної моделі розроблено та реалізовано алгоритм числового розрахунку мінімальної оцінки часу виробництва:

– знайти мінімальний час переналаштувань методом гілок і меж;

– перетворити отриману матрицю шляхом сортування та заміни елементів;

– знайти оптимальний план виробництва керуючись розв'язаною задачею комівояжера.

Необхідно відзначити, що в даній праці деталізовано описано методи знаходження мінімальних оцінок гілок поділу на кожному етапі гілкування, а також побудову плану виробництва по розв'язку задачі комівояжера. Побудований алгоритм дозволив вирішувати широкий клас задач теорії оптимізації випуску продукції. Практична цінність алгоритму у зменшенні обчислювальних потужностей для цього класу задач, що дасть можливість вирішувати ширші задачі спираючись на розв'язану.

Дослідження теорії планування виробництва показує, що при вирішенні широких класів задач застосування методів прямого перебору є дуже неефективне із збільшенням числа вершин. Методи гілок і меж дозволяють значно скоротити кількість обчислень, але для більш широких класів задач постає проблема обчислювальної складності алгоритму. Тому можливий перехід до евристичних методів розв'язання задачі комівояжера.

1. 1. R. Matai, S.P. Singh, M.L. Mittal, "Traveling Salesman Problem: An Overview of Applications,Formulations, and Solution Approaches ", Jaipur, India, 2010
2. R. Bosch and A. Herman,"Continuous line drawings via the traveling salesman problem," Operations Research Letters 3 (2004) 302-303,
3. Р. Базилевич, Р. Кутельмах // Комп'ютерні науки та інформаційні технології [Текст] :[зб. наук. пр.] / відп. ред. Ю. М. Рашкевич. – Л. : Видавництво Національного університету "Львівська політехніка", 2009. – 279 с. : іл. – (Вісник / Національний університет "Львівська політехніка"; № 650). – С. 235–244.
4. E. Nood and J. Been, An Efficient Transformation of the Generalized Travaling Salesman Problem, October 1991.
5. D.S. Johnson and L.A. McGeoch, "The Traveling Salesman Problem: A Case Study in Local Optimization", November 20, 1995.
6. Keramas James G. Robot technology fundamentals. – New York, Delmar Publishers, 1999. – 408 p.
7. Кирилович В.А. Автоматизоване формування маршрутів обслуговування робочих позицій промисловими роботами / В.А. Кирилович, О.В. Підтиченко //Вісник ТДТУ. – 2008. – Том. 13. – №4. – С. 152 – 157.
8. Шишмарёв В.Ю. Автоматизация производственных процессов в машиностроении. – М.:Издательский центр «Академия», 2007.
9. Полетаев В.А. Разработка компоновки и планировки гибких производственных систем: Методические указания. - Иваново: ИГЭУ, 1999.
10. Калин О.М., Ямпольский С.Л., Песков Л.В. Моделирование гибких производственных систем. – К.: Техника, 1991
11. Лаздынь С.В., Секирин А.И., Коробкова Т.А. Оптимизация компоновки технологического оборудования гибких производственных систем с использованием генетических алгоритмов. //Международный сборник научных трудов "Прогрессивные технологии и системы машиностроения", вып. 34. − Донецк: ДонНТУ, 2007. − С. 114-120
12. Курейчик В.В. Эволюционные методы решения оптимизационных задач. − Таганрог: Издво ТРТУ, 1999.

УДК 004.77
Шитий Д.В.
Національний технічний університет України «Київський політехнічний інститут»

СПОСІБ РЕЗЕРВНОГО КОПІЮВАННЯ ГОСТЬОВИХ ОПЕРАЦІЙНИХ СИСТЕМ В HYPER-V

Шитий Д.В. Спосіб резервного копіювання гостьових операційних систем в Hyper-V. У статті аналізуються складові частини віртуальних машин в Hyper-V. Представлено метод резервного копіювання гостьових операційних систем. Наведено інформацію про вміст *.ps1 та *.bat скриптів. Пропонується варіант розширення можливостей скрипту.
Ключові слова: Hyper-V, PowerShell, резервне копіювання, пакетний файл, VHD.

Shytyi D.V. **The way of backup guest operating systems in Hyper-V.** The article presents the composition of virtual machines in Hyper-V. The method of guest operating systems backup proposed. The article includes the information about the content of * .ps1 and * .bat scripts. Possible options to enhance the functions of the script proposed.
Keywords: Hyper-V, PowerShell, Backup, batch file, VHD.

Шитый Д.В. Способ резервного копирования гостевых операционных систем в Hyper-V. В статье анализируются составные части виртуальных машин в Hyper-V. Представлен метод резервного копирования гостевых операционных систем. Приведена информация о содержании * .ps1 и * .bat скриптов. Предлагается вариант расширения возможностей скрипта.
Ключевые слова: Hyper-V, PowerShell, резервное копирование, пакетный файл, VHD.

Вступ

Апаратна віртуалізація широко використовується майже в кожній частині IT-індустрії. Причин використання апаратної віртуалізації досить багато.

До них відносять наступні:

- консолідація, при якій багато невеликих фізичних серверів замінюються більш потужним сервером для збільшення ефективності використання апаратних ресурсів та зменшення використання електроенергії;
- спрощення процедури контролю функціонування ВМ;
- можливість встановлення додаткової ВМ без потреби закупівлі апаратного забезпечення;
- можливе перенесення ВМ з одного фізичного серверу на інший.

Апаратна віртуалізація є однією з системних вимог платформи апаратної віртуалізації Hyper-V. Система апаратної віртуалізації Hyper-v є безкоштовною, де всі функції доступні користувачу. Далі пропонується спосіб резервного копіювання гостьових операційних систем програмного продукту Hyper-V, який входить до складу Windows Server 2008.

Постановка наукової проблеми

Існують аналоги автоматичного резервного копіювання віртуальних машин в Hyper-V, але необхідні функції відсутні у безкоштовних версіях програмних продуктів. Задача полягає в розробці способу автоматичного резервного копіювання гостьових операційних систем (ОС) на Windows Server 2008 та платформі апаратної віртуалізації Hyper-V.

Виклад основного матеріалу та обґрунтування отриманих результатів дослідження

Віртуальна машина складається з наступинх файлів: конфігурації, експорту, віртуального диску, диференціальних дисків та файлів з розширенням *.BIN, *.VSV.

Файл конфігурації – файл, що зберігає налаштування віртуальної машини.

Файл експорту – файл з розширенням *.EXP в який записуються параметри з файлу конфігурації під час експорту віртуальної машини.

Файл віртуального диску – файл з розширенням *.VHD, що використовується як жорсткий диск [5].

Диференціальні диски – файли з розширенням *.AVHD. При створенні знімку гостьової операційної системи всі наступні зміни записуються у новий диск. що створюється у цей момент [5].

Файли розширенням *.BIN, *.VSV – файли, що створюються під час збереження стану віртуальної машини. Вся інформація з пам'яті віртуальної машини, вміст регістрів тощо записується в файли з розширенням *.BIN, *.VSV, і віртуальна машина переходить у стан «Вимкнута». Після увімкнення можна продовжити роботу з моменту збереження стану.

Отже, Hyper-V надає можливість створення миттєвих знімків (Snaphots). Але миттєві знімки лише забезпечують можливість повернення в відповідний часовий проміжок при виникненні помилки. Використовувати їх для резервного відновлення неможливо. Існують аналоги автоматичного резервного копіювання, такі як Veeam [7], які, однак, не дозволяють виконувати резервне копіювання за графіком в безкоштовних версіях Veeam, а отже таке резервне копіювання не буде автоматичним.

Спосіб автоматичного резервного копіювання

Для досягнення поставленої мети створюється файли запуску (скрипти), які реалізують наступні функції:

1) переведення віртуальної машини у стан «Вимкнено»;
2) копіювання складових частин віртуальної машини;
3) переведення віртуальних машин у стан «Увімкнено»;
4) впорядковування резервних копій за датою створення;
5) автоматичний запуск.

Заміна пункту 2 на пункт "переведення віртуальної машини у «Збереження стану»" дозволить продовжити роботу віртуальної машини з моменту переходу у «Збереження стану», що зменшить час, який було б витрачено на завантаження системи та необхідного програмного забезпечення.

Розглянемо більш детально визначені функції.

Переведення віртуальної машини в стан «Вимкнено»

Команда «Stop-VM 'Ubuntu Training Server' –Force» виконує процедуру виключення гостьових операційних систем в Hyper-V Manager. Можливо також перевести віртуальну машину у «Збереження стану» командою «Stop-VM 'Ubuntu Training Server' –Save –Force». Параметр «–Force» відповідає за те, що Hyper-V очікує деякий час, доки програми збережуть дані, а потім примусово завершає роботу.

Копіювання складових частин віртуальної машини

Командою «Copy-Item -R -Path 'D:\Virtual Machines\Ubuntu Training Server\' -Destination E:\VMs_Backup_by_Shitiy\$newfoldername» необхідні дані рекурсивно копіюються з джерела у місце призначення.

Переведення віртуальної машину в стан «Увімкнено»

Команда «Start-VM 'Ubuntu Training Server'» запускає віртуальну машину

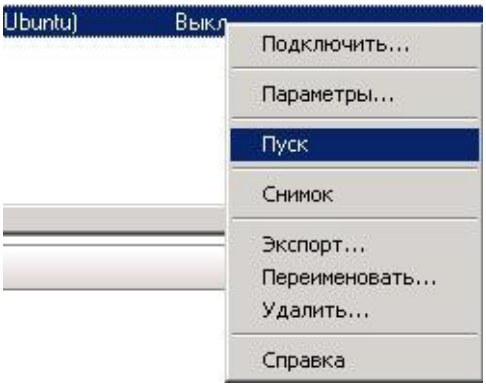


Рис. 1. Запуск віртуальної машини

Впорядковування резервних копій за датою створення

Впорядкування копій за датами здійснюється за допомогою створення теки кожного дня, в який виконувалось резервне копіювання.

Далі приведено вміст файлу Backup.ps1 :

```
$CurrentDate = Get-Date
gci E:\VMs_Backup_by_Shitiy | sort LastWriteTime | select -last 1 |
ForEach {$Date = $_.LastWriteTime}
    if ($Date.Day -lt 10) {$oday = "0" + $Date.Day}
    else {$oday = $Date.Day}
    if ($Date.Month -lt 10) {$omonth = "0" + $Date.Month}
    else {$omonth = $Date.Month}
    if  ($CurrentDate.Day -lt 10) {$nday = "0" + $CurrentDate.Day}
    else {$nday = $CurrentDate.Day          }
    if  ($CurrentDate.Month -lt 10) {$nmonth = "0" + $CurrentDate.Month}
    else {$nmonth = $CurrentDate.Month}
$nfoldername = [string]$oday +"."+ [string]$omonth +"."+ [string]$Date.Year + " - " + [string]$nday +"."+
[string]$nmonth +"."+ [string]$CurrentDate.Year
New-Item E:\VMs_Backup_by_Shitiy -Name $nfoldername -Type Directory
```

Приведена частина скрипту сортує вміст вказаної теки та створює нову теку з проміжком часу.

Автоматичний запуск

Для автоматичного запуску скрипту з розширенням *.ps1 (Powershell) необхідно створити скрипт з розширенням *.bat (CMD), оскільки саме *.bat виконується через планувальник завдань, що входить до Windows Server 2008r2.

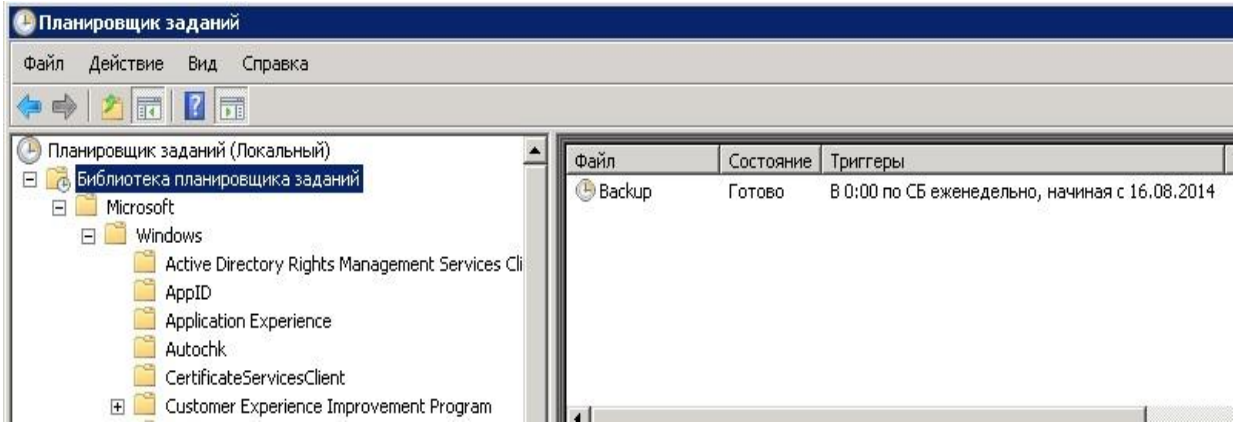


Рис. 2. Завдання Backup у планувальнику завдань

Типово PowerShell не має командлетів для управління Hyper-v. Тому в *.bat скрипті імпортуємо модуль для роботи з Hyper-V. Модуль Hyper-V необхідно завантажити та встановити перед використанням [1]. Розроблений скрипт з розширенням *.ps1 запускаємо в PowerShell.

Скрип з розширенням *.bat має наступний вміст:

```
1  %SystemRoot%\system32\WindowsPowerShell\v1.0\powershell.exe -noExit CRLF
2  -Command " Import-Module '%ProgramFiles%\modules\hyperV'; CRLF
3  C:\Backup_settings_by_Shitiy\Backup.ps1;"
```

Рис. 3. Вміст скрипту з розширенням *.bat

Висновки

Запропонований спосіб резервного копіювання призводить до певного простою обладнання, а саме: в момент виконання збереження стану віртуальної машини та створення резервної копії необхідних файлів. Розроблений скрипт для автоматичного резервного копіювання з використанням оболонки PowerShell. Можливе розширення функцій скрипту. Модифікація може полягати у створення резервної копії на віддаленій робочій станції тощо.

Існують аналоги автоматичного резервного копіювання віртуальних машин в Hyper-V, але необхідний функціонал відсутній у безкоштовних версіях. Саме тому запропонований спосіб наразі є актуальним рішенням задачі. Наприклад, автоматичне резервне копіювання з варіантом переведення віртуальної машини у стан «Вимкнено» можна виконувати для контролерів домену.

1. *James O'Neill.* A PowerShell Module for Hyper-V. / James O'Neill // Режим доступу: http://pshyperv.codeplex.com/downloads/get/101935#
2. *Попов А.В.* Введение в Windows PowerShell. / *Попов А.В.* // Режим доступу: https://vk.com/doc240060454_287571866?hash=4d8ab18b95c98 3c349&dl=b7ea60ddf2b7ede82d
3. *А. Н. Чекмарев.* Microsoft Windows 2000: Server и Professional. / А. Н. Чекмарев, Д. Б. Вишнякова. // Режим доступу: http://www.google.ru/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0CBwQFjAA&url=http%3A%2F%2Fpsbatishev.narod.ru%2Fbooks%2F00165.doc&ei=JmdaVcLRDYqaygOyuoCgBw&usg=AFQjCNHJjMJHZ08ztSZAG44dtJdnSTUdyQ&bvm=bv.93564037,d.bGQ
4. *В.Г. Казаков.* Технологии и алгоритмы копирования. / В.Г. Казаков, С.А. Федосин // Режим доступу: http://www.ict.edu.ru/ft/005653/62330e1-st17.pdf
5. *Morimoto Rand.* Windows Server 2008 Hyper-V Unleashed. / R. Morimoto, Jeff Guillet. // Режим доступу: http://edc.tversu.ru/elib/inf/0260.pdf
6. *Microsoft.* Windows Server 2008 Active Directory. / Microsoft // Режим доступу: https://technet.microsoft.com/en-us/library/dd448614.aspx
7. *Veeam.* Veeam Backup & Replication v8. / Veeam. / Режим доступу: http://www.veeam.com/ru/backup-version-standard-enterprise-editions-comparison.html
8. *Vic Laurie.* The Command Line in Windows / V. Laurie. // Режим доступу: http://www.didiermorandi.fr/vbscript/doc/Windows_Command_Line_Vic_Laurie.pdf
9. *Idera* . Mastering Powershell. / Idera . // http://powershell.com/Mastering-PowerShell.pdf
10. *Jean Ross.* The Windows PowerShell Owner's Manual / J. Ross., G. Stemp // Режим доступу: https://allunifiedcom.files.wordpress.com/2010/07/powershell_v2_owners_manual.pdf

УДК 004.312
Шолом П.С., Семенюк В.Я., Бєляков О.В.
Луцький національний технічний університет

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ НАЛАГОДЖУВАЛЬНОЇ ПЛАТИ ARDUINO-LITE-KIT

Шолом П.С., Семенюк В.Я., Бєляков О.В. Програмне забезпечення налагоджувальної плати ARDUINO-LITE-KIT. Розроблено демонстраційне програмне забезпечення для апаратно-програмного комплексу ARDUINO-LITE-KIT. Даний набір можна використовувати у навчальному процесі для набуття навичок роботи як з програмним, так і апаратним забезпеченням.
Ключові слова: програмне забезпечення, Arduino Uno R3, навчальний набір ARDUINO-LITE-KIT

Шолом П.С., Семенюк В.Я., Беляков А.В. Программное обеспечение отладочной платы ARDUINO-LITE-KIT. Разработано демонстрационное программное обеспечение для аппаратно-программного комплекса ARDUINO-LITE-KIT. Данный набор можно использовать в учебном процессе для приобретения навыков работы как с программным, так и аппаратным обеспечением.
Ключевые слова: программное обеспечение, Arduino Uno R3, учебный набор ARDUINO-LITE-KIT

Sholom P.S., Semenyuk V.Y., Beliakov O.V. Software for the development board ARDUINO-LITE-KIT. Demo software for hardware-software complex ARDUINO-LITE-KIT is developed. This kit can be used in the educational process for learning to work with both software and hardware.
Keywords: software, Arduino Uno R3, training set ARDUINO-LITE-KIT

Постановка проблеми. Сучасне суспільство вимагає від людини володіння інформаційними технологіями, а одним із головних складових інформаційних технологій є програмне та апаратне забезпечення. Тобто сучасний учень / студент повинен вивчати і програмну, і апаратну складові інформаційних технологій. Але вивчення часто означає лише отримання знань, в кращому випадку, навичок в галузі програмного забезпечення. Оснащення освітнього процесу відповідно до змісту навчальних предметів є актуальною проблемою на даний час [1]. Для вирішення перерахованих проблем можна скористатися вільно поширюваним апаратно-програмним комплексом Arduino.

Аналіз останніх досліджень і публікацій. Даною тематикою займались Болл Стюарт Р. [2], Гололобов В.Н., Кравченко А.В. [3], Петін В.А. [4], Уилли Соммер [5]. Зокрема в [5] розглянуто програмування мікроконтролерних плат Arduino / Freduino; описана структура і функціонування мікроконтролерів, середовище програмування Arduino, необхідні інструменти та комплектуючі для проведення експериментів; докладно розглянуті основи програмування плат Arduino: структура програми, команди, оператори та функції, аналоговий і цифровий ввід / вивід даних; виклад матеріалу супроводжується більше 80 прикладами з розробки різних пристроїв: реле температури, шкільного годинника, цифрового вольтметра, сигналізації з давачем переміщення, вимикача вуличного освітлення тощо. В [1] розглядається проблема впровадження інформаційних технологій у навчальний шкільний процес на базі плат Arduino.

Метою роботи є розробка та верифікація демонстраційного програмного забезпечення для учбово-лабораторного стенду (УЛС) на базі набору ARDUINO-LITE-KIT.

Виклад основного матеріалу роботи. Arduino − апаратна обчислювальна платформа, основними компонентами якої є плата вводу / виводу та середовище розробки на мові Processing / Wiring. Arduino може використовуватися як для створення автономних інтерактивних об'єктів, так і підключатися до програмного забезпечення, яке виконується на комп'ютері (наприклад: Adobe Flash, Processing, Max / MSP, Pure Data, SuperCollider). Інформація про плату (малюнок друкованої плати) знаходиться у відкритому доступі і може бути використана тими, хто вважає за краще збирати плати самостійно.

Плата Arduino складається з мікроконтролера Atmel AVR, а також елементів обв'язки для програмування та інтеграції з іншими пристроями. На багатьох платах наявний лінійний стабілізатор напруги +5 В або +3,3 В. Тактування здійснюється на частоті 16 або 8 МГц кварцовим резонатором. У мікроконтролер записаний завантажувач (bootloader), тому зовнішній програматор не потрібен.

На концептуальному рівні усі плати програмуються через RS-232 (послідовне з'єднання), але реалізація даного способу різниться від версії до версії. Новіші плати програмуються через

USB, що можливо завдяки мікросхемі конвертера USB-to-Serial FTDI FT232R. У версії платформи Arduino Uno в якості конвертера використовується контролер Atmega8 у SMD-корпусі. Дане рішення дозволяє програмувати конвертор таким чином, щоб платформа відразу розпізнавалась як миша, джойстик чи інший пристрій за вибором розробника зі всіма необхідними додатковими сигналами керування. У деяких варіантах, таких як Arduino Mini або неофіційній Boarduino, для програмування потрібно підключити до контролера окрему плату USB-to-Serial або кабель.

Плати Arduino дозволяють використовувати значну кількість I/O виводів мікроконтролера у зовнішніх схемах. Наприклад, у платі Decimila доступно 14 цифрових входів/виходів, 6 із яких можуть видавати ШІМ сигнал, і 6 аналогових входів. Ці сигнали доступні на платі через контактні площадки або штирьові розніми. Також існує декілька видів зовнішніх плат розширення, які називаються «shields», які приєднуються до плати Arduino через штирьові роз'єми.

У даній роботі описується робота із базовим набором ARDUINO-LITE-KIT на основі спрощеної версії Arduino UNO R3. Розглядається робота із периферійними пристроями, зокрема такими, як пульт дистанційного керування за допомогою інфрачервоного зв'язку. Для цього було використано мікросхему інфрачервоного давача Infrared Receiver IC 38 KHz (VS-1838) та пульт дистанційного керування IR Infrared Remote Control Kit 2 (SE-020401). Розроблено демонстраційне програмне забезпечення.

Навчальний набір ARDUINO-LITE-KIT (рис. 1) на базі Arduino UNO R3 − це електронний конструктор і зручна платформа швидкої розробки електронних пристроїв. Платформа користується величезною популярністю в усьому світі завдяки зручності і простоті мови програмування, а також відкритій архітектурі і програмному коду. Пристрій програмується через USB без використання програматорів.

Завдяки великому асортименту допоміжних модулів та давачів набір ARDUINO-LITE-KIT є цікавим як початківцям-електронникам, так і професіоналам. Набір створений на основі плати ARDUINO-UNO ревізії R3.

Таблиця 1. **Технічні характеристики плати Arduino Uno R3**

Мікроконтролер:	ATmega328
Робоча напруга:	5V
Вхідна напруга:	(Рекомендується) 7-12V
Вхідна напруга:	(Межі) 6-20V
Цифрові вводи / виводи:	14 (6 з яких ШИМ)
Аналогові входи:	6
Постійний струм в лінії вводу / виводу:	40 мА
Постійний струм на 3,3V Pin:	50 мА
Флеш-пам'ять:	32 Кб (ATmega328),
	0.5 Кб використовуються завантажувачем
SRAM	2 Кб (ATmega328)
EEPROM	1 Кб (ATmega328)
Тактова частота	16 МГц

Комплектація набору:
1. Плата Arduino Uno R3.
2. USB-кабель.
3. LED дисплей на 1 цифру (2 шт.).
4. LED дисплей на 4 цифри.
5. 8x8 точок екран-матриця.
6. Мікросхема 74HC595N (восьмибітний регістр з послідовним і паралельним виходом).
7. Динамік-пищалка.
8. Давач температури LM35.
9. Фоторезистор (3шт.).
10. Давач температури і вологості.
11. Конденсатори електролітичні (2 шт.).
12. Інфрачервоний приймач.

13. Набір резисторів (20 шт.).
13. Світлодіоди (сині, зелені, червоні − по 5 шт.).
14. Макетна плата MB-102.
15. Тактові кнопки з ковпачками (4 шт.).
16. Потенціометр (змінний резистор).
17. Набір проводів різної довжини і конфігурації.
18. Штирьовий роз'єм на 40 контактів.
19. Батарейний відсік на 6 елементів (тип AA) з можливістю підключення до ARDUINO.
2 0. Інфрачервоний пульт ДК.
21. Трибарвний RGB модуль.

Рис. 1. **Зовнішній вигляд набору ARDUINO-LITE-KIT**

Для моделювання пристрою ІЧ-давача використано наступні електронні компоненти: плата Arduino Uno R3, USB-кабель, з'єднувальні провідники, інфрачервоний давач та пульт ДК (рис. 2)

Рис. 2. **Електронні компоненти для макетування ІЧ-давача**

Схема підключення ІЧ-давача VS1838B до плати Arduino Uno зображено на рисунку 3. Можливі варіанти із живленням +5 В та +3,3 В, що допускається згідно специфікації мікросхеми давача.

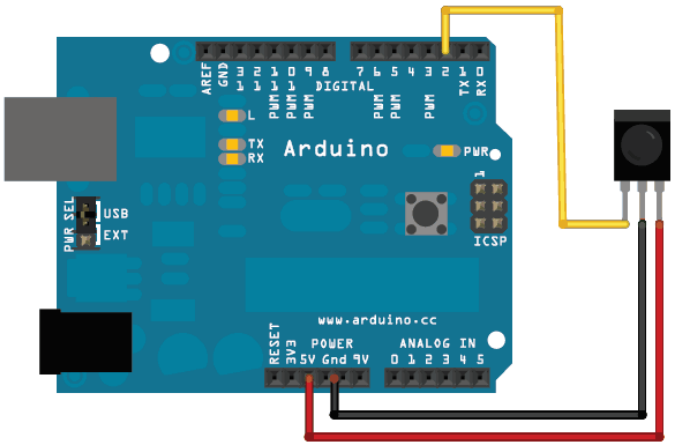


Рис. 3. **Схема підключення ІЧ-давача VS1838B**

Freeduino / Arduino програмується на спеціальній мові програмування. Вона базується на C / C++ і дозволяє використовувати будь-які його функції. Строго кажучи, окремої мови Arduino не існує, як і не існує компілятора Arduino. Написані програми перетворюються (з мінімальними змінами) в програму мовою C / C ++ і потім компілюються компілятором AVR-GCC. Отже, фактично, використовується спеціалізований для мікроконтролерів AVR варіант C / C++. Різниця полягає в тому, що розробник отримує просте середовище розробки і набір базових бібліотек, що спрощують доступ до периферії, яка знаходиться на одній платі із мікроконтролером.

При читанні або записі до цифрового порту застосовується лише два можливих значення – порт може бути встановлений як HIGH (високий рівень) або LOW (низький рівень). Рівень HIGH відповідає 5 В на виході. При читанні значення на цифровому порті, починаючи з 3 В і вище, мікропроцесор сприйме цю напругу як HIGH. Ця константа представлена цілим числом 1. Рівень LOW відповідає 0 В на виході порту. При читанні значення на цифровому порті, починаючи з 2 В і менше, мікропроцесор сприйме цю напругу як LOW. Ця константа представлена цілим числом 0.

Цифрові порти можуть використовуватися на ввід або вивід сигналів. Зміна порту з вводу на вивід виробляється за допомогою функції pinMode (). Порти, сконфігуровані на введення сигналів, мають великий вхідний опір, що дозволяє підключати до них джерело сигналу, і порт не буде споживати великий струм. Порти, сконфігуровані на вивід сигналів, мають малий вихідний опір. Це означає, що такі порти можуть забезпечувати підключені до них елементи електроенергією. У цьому стані порти підтримують позитивний чи негативний напрям струму до 40 мА (міліампер) на інші пристрої або схеми. Це дозволяє підключити до них будь-яке навантаження, наприклад світлодіод (через резистор, що обмежує струм). Порти, сконфігуровані як виводи, можуть бути пошкоджені, якщо їх замкнути накоротко на «землю» (загальна шина живлення), на джерело живлення +5 В або під'єднати до потужного навантаження з малим опором.

Freeduino має вбудований контролер для послідовної передачі даних, який може використовуватися як для зв'язку між Freeduino / Arduino пристроями, так і для зв'язку з комп'ютером. На комп'ютері відповідне з'єднання представлено USB COM-портом. Зв'язок відбувається по цифрових портах 0 і 1, і тому не можна використовувати їх для цифрового вводу / виводу якщо використовувати функції послідовної передачі даних.

Нижче представлено опис основних бібліотек, використаних при розробці ПЗ.

Бібліотека Servo. Ця бібліотека для Arduino-контролера надає набір функцій для управління сервоприводами. Стандартні сервоприводи дозволяють повертати привід на визначений кут від 0 до 180 градусів зазвичай. Деякі сервоприводи дозволяють здійснювати повні оберти на заданій швидкості. Бібліотека Servo дозволяє одночасно керувати 12-ма сервоприводами на більшості плат Arduino і 48-ма на Arduino Mega. На контролерах, відмінних від Mega, використання бібліотеки відключає можливість використовувати виходи 9 і 10 в режимі ШІМ навіть якщо привід не підключений до цих виводів. На платі Mega можуть бути використані до 12

сервоприводів без втрати функціоналу ШІМ. При використанні Mega для управління від 12 до 23 сервоприводів не можна буде використовувати виходи 11 і 12 для ШІМ.

Функції бібліотеки: attach(), write(), writeMicroseconds(), read(), attached(), detach().

У загальному випадку сервопривід підключається 3-ма проводами: живлення, земля і сигнальний. Зазвичай за живлення відповідає червоний провід і може бути підключений до виводу + 5V на платі Arduino. Чорний провід «земля» підключається до GND виводу Arduino. Сигнальний провід зазвичай жовтий і провід підключається до цифрового виводу контролера Arduino. Слід зазначити, що потужні сервоприводи можуть створювати велике навантаження. В цьому випадку він повинен живитися окремо (не через вихід + 5V Arduino). Теж саме стосується і випадку підключення відразу декількох сервоприводів. Слід переконатися, що привід і контролер підключені до загальної землі.

Бібліотека EERPOM. Мікроконтролери ATmega мають свою незалежну пам'ять, тобто у користувачів Arduino є можливість зберігати дані в цій пам'яті і вони можуть бути використані після увімкнення / вимкнення або перезавантаження контролера. Arduino-бібліотека EERPOM надає зручний і простий інтерфейс роботи з цією пам'яттю. Різні моделі мікроконтролерів відрізняються обсягом EERPOM- пам'яті (наприклад, 1024 байт у ATMega328, 512 байт у ATmega168 та ATmega8 і по 4Кб (4096 байт) у ATmega1280 і ATmega2560).

Функції бібліотеки: read(), write().

Бібліотека SPI. Бібліотека SPI дозволяє контролеру Arduino взаємодіяти з пристроями, що підтримують SPI-протокол. Arduino в даному випадку виступає в якості ведучого пристрою. Послідовний периферійний інтерфейс (SPI) – це послідовний синхронний протокол передачі даних, що використовується мікроконтролерами для обміну даними з одним або декількома периферійними пристроями на невеликих відстанях. Для організації з'єднання SPI необхідний однопровідний пристрій. Зазвичай це мікроконтролер, який управляє з'єднанням з веденими пристроями. Підключення здійснюється трьома загальними лініями і лінією вибору периферійного (веденого) пристрою: Master In Slave Out (MISO), Master Out Slave In (MOSI), Serial Clock (SCK), Slave Select pin. Якщо вході останнього з перерахованих ліній LOW, то ведений взаємодіє з ведучим, якщо HIGH, то ведений ігнорує сигнали від ведучого.

При роботі з SPI пристроями треба враховувати такі моменти: порядок виведення даних, рівень сигналу синхронізації, фаза синхронізації.

Виробники SPI пристроїв дещо по різному реалізують протокол, тому необхідно уважно ознайомитися з технічним описом до пристрою.

Функції бібліотеки: begin(), end(), setBitOrder(), setClockDivider(), setDataMode(), transfer().

Бібліотека Stepper. Бібліотека надає зручний інтерфейс управління біполярними і уніполярними кроковими двигунами.

Розроблено таке програмне забезпечення:

1. Програма зчитування кодів натиснутих клавіш ІЧ-пульта керування. Програма базується на вільно розповсюджуваній бібліотеці IRremote. За допомогою даної програми визначено коди клавіш ІЧ-пульта.

2. Програма «Цілочисельний безпровідний калькулятор». Суть роботи даного ПЗ полягає у використанні клавіатури пульта IR Infrared Remote Control Kit 2 (SE-020401) в ролі клавіатури калькулятора, а роль дисплея для відображення результатів відіграє консоль COM-порта комп'ютера.

Висновки. Розроблено демонстраційне програмне забезпечення для апаратно-програмного комплексу ARDUINO-LITE-KIT. Даний набір можна використовувати у навчальному процесі для набуття навичок роботи як з програмним, так і апаратним забезпеченням.

1. Практическое занятие с использованием микро-ЭВМ «Ардуино» [Електронний ресурс]. − Режим доступу: http://konkurs.ciur.ru/методическая-копилка-работ-победите/
2. Болл Стюарт Р. Аналоговые интерфейсы микроконтроллеров. − М.: Издательский дом «Додэка-XXI», 2007. − 360 с.
3. Кравченко А.В. 10 практических устройств на AVR-микроконтроллерах. Книга 2. СПб.: «КОРОНА-ВЕК», 2009. − 320с.
4. Петин В. А. Проекты с использованием контроллера Arduino. − СПб.: БХВ-Петербург, 2014. − 400 с.
5. Уилли Соммер. Программирование микроконтроллерных плат Arduino/Freeduino. − СПб.: БХВ-Петербург, 2012. − 256 с.

УДК 514.18
Бадаєв Ю.І., Ганношина І.М.
Київська державна академія водного транспорту ім. гетьмана Петра Конашевича-Сагайдачного (КДАВТ)

МОДЕЛЮВАННЯ NURBS – КРИВИХ НА ОСНОВІ ПРИНЦИПУ ДВОЇСТОСТІ

Бадаєв Ю.І., Ганношина І.М. Моделювання NURBS-кривих. В статті пропонується новий метод проектування кривої, яка є огинаючою сім'ї відрізків прямих, визначених на основі NURBS-технології.
Ключові слова: NURBS-криві, огинаюча, кривина.

Бадаев Ю.И., Ганношина И.Н., Моделирование NURBS - кривых на основе принципа двойственности. В статье предлагается новый метод проектирования кривой, которая является огибающей семьи отрезков прямых, определенных на основе NURBS-технологии.
Ключевые слова: NURBS-кривые, огибающая, кривизна.

Badaev Y., Gannoshina I. Modeling NURBS- curves based on the principle of duality. The paper proposes a new design method of the curve which is the envelope of the family of straight segments defined on the basis of NURBS technology.
Keywords: NURBS-curves, the envelope, the curvature.

Постановка проблеми. В роботі проектувальників обводів, які працюють в рухомому середовищі, нерідко виникає задача проектування кривої, яка має наперед заданий закон зміни кривини.

Відомі методи не дають змоги розв'язувати поставлену задачу, тому необхідно створити нові підходи у розв'язанні цієї задачі.

В даній роботі пропонується проектування спеціальної кривої як огинаючої сім'ї відрізків прямих.

Такий підхід дає змогу в подальшому аналізувати закон зміни цих прямих, які є дотичними до кривої, що проектується. Вважаючи на те, що ці відрізки прямих задають перші похідні, то аналізуючи закон їх зміни, можна розв'язати задачу проектування кривої за заданим законом зміни кривини.

Аналіз останніх досліджень і публікацій. В останніх публікаціях [1-7] даються методи моделювання різних кривих, але не зазначаються підходи до аналізу їх форми і кривини уздовж кривої.

Метою даної статті є розробка методу проектування огинаючої сім'ї відрізків прямих, які визначаються на основі NURBS-технологій, що дає можливість аналізувати форму і закон зміни кривини.

Основна частина. Як відомо із проективної геометрії [1], пряму

$$ax + bx + c = 0 \tag{1}$$

можна розглядати, як точку із координатами a,b,c.

Застосуємо до цих координат формули визначення NURBS – кривих [2]:

$$a = a(t) = \frac{\sum_{i=1}^{n} N_i, m(t) w_i a_i}{\sum_{i=1}^{n} N_i, m(t) w_i}, \tag{2}$$

$$b = b(t) = \frac{\sum_{i=1}^{n} N_i, m(t) w_i b_i}{\sum_{i=1}^{n} N_i, m(t) w_i}, \tag{3}$$

$$c = c(t) = \frac{\sum N_i, m(t) w_i c_i}{\sum N_i, m(t) w_i} \tag{4}$$

де n – кількість заданих прямих, $N_i, m - B$ – сплайн по рядку m ..

Підставимо (3), (4) в (1). Отримаємо рівняння сім'ї прямих

$$a(t)x + b(t)y + c(t), \tag{5}$$

яке визначає її положення в сім'ї, де t – параметр сім'ї.

До цієї сім'ї прямих можна провести огинаючу криву, яка буде відмінна від кривої, що визначається за векторним рівнянням NURBS – кривої. Огинаюча крива визначиться системою [3]:

$$\begin{cases} f = \mathrm{a}(\mathrm{t})\mathrm{x} + \mathrm{b}(\mathrm{t})\mathrm{y} + \mathrm{c}(\mathrm{t}) = 0, \\ \dfrac{df}{dt} = a'(\mathrm{t})\mathrm{x} + \mathrm{b}'(\mathrm{t})\mathrm{y} + \mathrm{c}'(\mathrm{t}) = 0. \end{cases} \tag{6}$$

Виключимо із цієї системи *x* і *y*. Для цього помножимо перше рівняння на і віднімемо друге. Будемо мати:

$$x = \frac{c''(t) - c(t)\dfrac{b'(t)}{b'(t)}}{a * \dfrac{b'(t)}{b'(t)} - a''(t)}. \tag{7}$$

Аналогічно виключимо *x*, помноживши перше рівняння на $\dfrac{a'(t)}{a'(t)}$.

Отримуємо:

$$y = \frac{c'(t) - c\dfrac{a'(t)}{a'(t)}}{b * \dfrac{a'(t)}{a'(t)} - b''(t)}. \tag{8}$$

Рівняння (7) і (8) визначають нову параметричну криву. Переваги отриманої кривої полягають у тому, що дотична в кожній точці T (x,y) отриманої огинаючої визначаються формулою (1).

Дотична буде визначати похідну в заданій точці. Аналіз закону зміни першої похідної дасть змогу прогнозувати форму кривої, а також закон зміни кривими, що є важливим в проектуванні обводів машин, які працюють у рухомому середовищі: обводи літаків, автомобілів, суден тощо.

Для підтвердження достовірності описаного методу виведемо формули огинаючої кривої на основі кривих Безьє 2-го і 3-го порядку.

Крива Безьє другого порядку задається формулою [3]:

$$r = r_0(1-t)^2 + 2r_0(1-t)t + r_1 t^3 \tag{9}$$

Крива Безьє третього степеня задається формулою [3]:

$$r = r_0(1-t)^3 + 3r_1(1-t)^2 t + 3r_2(1-t)t^2 + r_{13}t^3 \quad (10)$$

Розкриємо дужки в формулах (9), (10).
Будемо мати:
Для кривої Безьє 2-го степеня:

$$r = r_0 + 2(r_1 - r_0)t + (r_1 - 2r_1 + r_2)t^2 \quad (11)$$

Для кривої Безьє 3-го степеня:

$$r = r_0 + 3(r_1 - r_0)t + (6r_1 - r_0)^2 + (r_3 r_0 + 3 - 3r_1)t^3 \quad (12)$$

похідна від (11) буде:

$$r' = 2(r_1 - r_0) \quad (13)$$

похідна від (12) буде:

$$r' = 3(r_1 - r_0) + 2(6r_1 - r_0)t + 3(r_3 - r_0 + 3r_2 - 3r_1)t^2 \quad (14)$$

Таким чином огинаюча на основі кривих Безьє 2-го і 3-го степенів визначаються формулами:

$$x = \frac{c' - c\dfrac{b}{b'}}{c_1\dfrac{b}{b'} - a_1}, \quad y = \frac{c' - c\dfrac{a}{a'}}{b\dfrac{a}{a'} - b'}, \quad (15)$$

Де:
Для кривих Безьє 2-го степеня буде:

$$a = a_0 + 2(a_1 - a_0)t + (a_0 - 2a_1 + a_2)t^2$$
$$a' = 2(a_1 - a_0)$$
$$b = b_0 + 2(b_1 - b_0)t + (b_0 - 2b_1 + b_2)t^2$$
$$b' = 2(b_1 - b_0)$$
$$c = c_0 + 2(c_1 - c_0)t + (c_0 + 2c_1 + c_2)t^2$$
$$c' = 2(c_1 - c_0).$$

Для кривої Безьє 3-го степеня буде:

$$a = a_0 + 3(a_1 - a_0)t + (6a_1 - a_0)t^2 + (a_3 - a_0 + 3a_2 - 3a_1)t^3$$
$$a' = 3(a_1 - a_0)t + 2(6a_1 - a_0)t + 3(a_3 - a_0 + 3a_2 - 3a_1)t^2$$
$$b = b_0 + 3(b_1 - b_0)t + (6b_1 - b_0)t^2 + (b_3 + 3b_2 - 3b_1)t^3$$
$$b' = 3(b_1 - b_0) + 2(6b_1 - b_0)t + 3(b_3 + 3b_2 - 3b_1)t^2$$
$$c = c_0 + 3(c_1 - c_0)t + (6c_1 - c_0)t^2 + (c_3 - c_0 + 3c_2 - 3c_1)t^3$$
$$c' = 3(c_1 - c_0)t + 2(6c_1 - c_0)t + 3(c_3 + 3c_2 - 3c_1)t^2.$$

В роботі розроблено комп'ютерну програму, робота якої представлена на рис. 1 і 2.

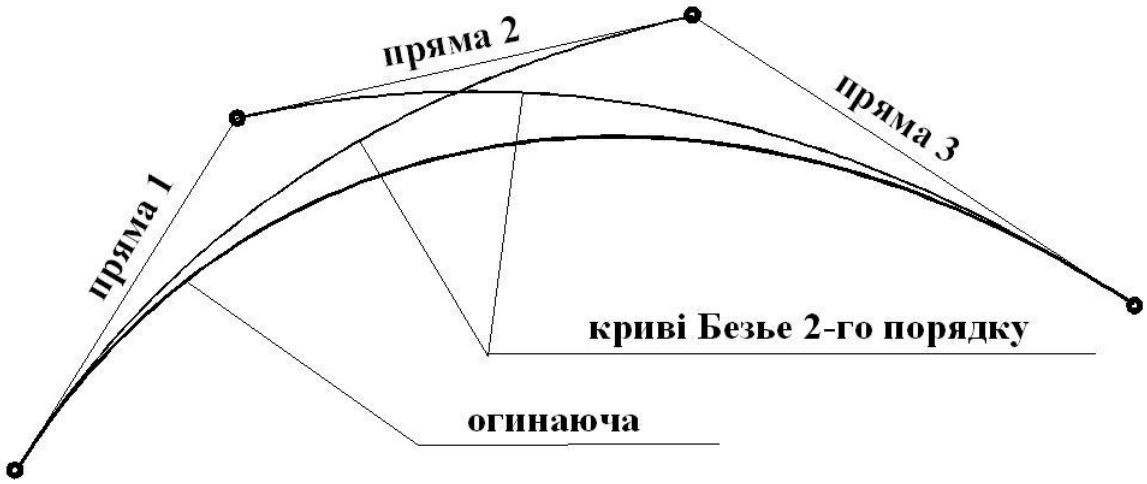


Рис. 1. **Огинаюча на основі двох кривих Безьє 2-го порядку**

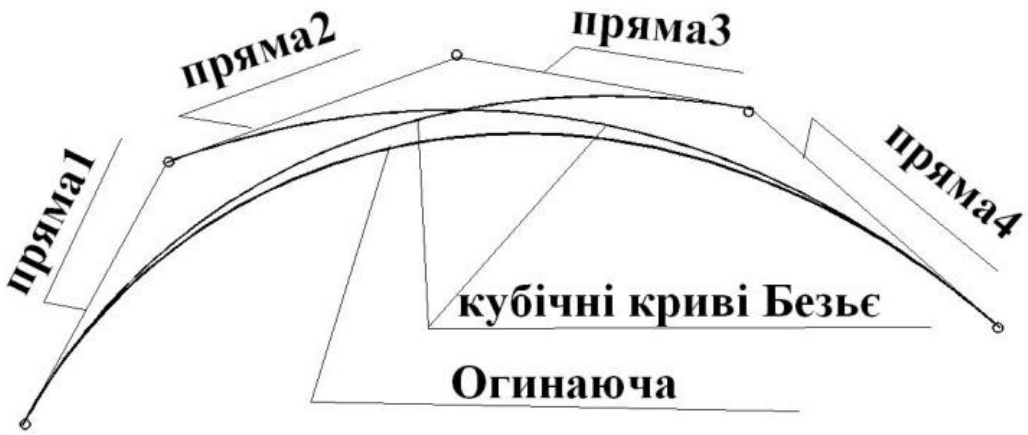


Рис. 2. **Огинаюча на основі двох кривих Безьє 3-го порядку**

Висновки. Таким чином, в статті пропонується новий метод проектування кривої, яка є огинаючою сім'ї відрізків прямих, визначених на основі NURBS-технології.

Подальші дослідження передбачається проводити щодо виявлення особливостей запропонованого методу.

1. Ефимов Н.В. "Высшая геометрия" – М.: Издательство "Наука" 1971-576с.
2. Фокс А. Пратт М. "Вычислительная геометрия. Применение в проектировании и на производстве": Переведено с английского – М.: Издательство "Мир", 1982-304с.
3. Голованов Н.М. "Геометрическое моделирование" – М.: Издательство физико – математической литературы; 2002 – 472с.
4. Гавриленко Є..А.Формування просторової дискретно-поданої кривої на основі прилягаючих кіл/ Сучасні проблеми моделювання: зб. наук. праць/ МДПУ ім. Б.Хмельнтцького, 2014-Вип. 3, с.39-43.
5. Савченко О.О. Геометричне моделювання профілів морських хвиль на основі трохоїдальної моделі / Сучасні проблеми моделювання: зб. наук. праць/ МДПУ ім. Б.Хмельнтцького, 2014-Вип. 3, с.78-87.
6. Аппроксимация методом Безье http://graphics.cs.msu.ru/grafor/gr_help/chapter_5_8.htm
7. Геометрическое сглаживание B-сплайнами http://www.codenet.ru/progr/alg/B-Splines/

УДК 515.2:536.3
Бакалова В.М., Лебедєва О.О., Юрчук В.П.
Національний технічний університет України «Київський політехнічний інститут»

ПОБУДОВА ТРИВИМІРНОЇ МОДЕЛІ ТА КОМПОНОВКА КРЕСЛЕНИКА ТІЛА СКЛАДНОЇ ФОРМИ В СЕРЕДОВИЩІ AUTOCAD 2015

Бакалова В.М., Лебедєва О.О., Юрчук В.П. Побудова тривимірної моделі та компоновка кресленика тіла складної форми в середовищі AutoCAD 2015. В статті розглянуто основні етапи побудови твердотільних моделей та компоновка їх креслеників у системі AutoCAD 2015.
Ключові слова: твердотільна модель, інформаційні технології, моделювання, середовище AutoCAD 2015.

Бакалова В.М., Лебедева О.О., Юрчук В.П. Построение трехмерной модели и компоновка чертежа тела сложной формы в среде AutoCAD 2015. В статье рассмотрены основные этапы построения твердотельних моделей и компоновка их чертежей в системе AutoCAD 2015.
Ключевые слова: твердотельная модель, информационные технологии, моделирование, среда AutoCAD 2015.

Bakalova V., Lebedeva O., Yurchuk V. Construction of a three-dimensional model and layout drawing body forms a complex in medium AutoCAD 2015. This article describes the main stages of formation of solid models in AutoCAD 2015.
Keywords: solid model, information technology, modeling, medium AutoCAD 2015.

Постановка проблеми. У світі сучасних інформаційних технологій, що розвиваються стрімкими темпами, високого рівня розвитку програмного і технічного забезпечення комп'ютерної техніки необхідно застосовувати нові високопродуктивні методи та засоби моделювання [1, 2]. Це вимагає пошук нових підходів для підвищення ефективності навчання. Тому, особливе значення набувають заходи впровадження нових методів у навчальний процес.

Аналіз останніх досліджень і публікацій. Сучасні процеси проектування та конструювання важко уявити без тривимірного моделювання об'єктів. Отже, тривимірне моделювання об'єктів широко використовується у багатьох галузях і широко висвітлене в літературі.

Невирішені частини проблеми. Однак в методичному плані педагогіки вищої школи має місце використання спрощеного матеріалу з основ інженерних дисциплін. А тому, потрібні нові підходи, засоби для розв'язання інженерних задач [3, 4].

Метою цієї роботи є використання комп'ютерних технологій для побудови тривимірних об'єктів та компоновки їх креслеників.

Основні результати дослідження. Застосування сучасних програм надає широкі можливості для побудови твердотільних моделей, дослідження їх, виконання складальних і робочих креслеників, автоматизувати процес параметричного моделювання.

Багато років одним з найбільш потужних і широко поширених інструментів проектування є система AutoCAD. У кожній новій версії можливості програми стають все ширше, елементи управління модернізуються, з'являються нові. Моделювання тривимірних об'єктів має певні переваги. По-перше, за допомогою програмного, технічного, методичного забезпечення проводити дослідження моделей. По-друге, за тривимірною моделлю створювати кресленики, уникнувши при цьому помилок. Твердотільні об'єкти мають складну форму, а тому побудова їх починається з формування твердотільних примітивів шляхом застосування теоретико-множинних операцій (об'єднання, віднімання, перетину та ін.) [1, 2, 5]. Для цього потрібні знання в області геометрії, стереометрії, математики, фізики, розуміння об'єму і форми, а також володіння основами архітектури, фотографії, дизайну і ін.

При моделюванні від задуму ідеалізованого об'єкту до отримання кінцевого продукту було запропоновано алгоритм створювання моделей в такій послідовності дій: дослідження об'єктів, побудова об'єктів, використання матеріалів, візуалізація, редагування, підготовка до друку.

Розглянемо створення тривимірної моделі і комплексного рисунку на прикладі, що наведений на рис. 1.

Основні етапи побудови 3D-моделі та компоновки кресленика.

Перший етап виконання пропонованої задачі – аналіз форми геометричних елементів, що утворюють зовнішню і внутрішню поверхні тіла.

Зовнішня поверхня:

- сфера діаметра 200 мм, зрізана двома горизонтальними площинами, симетрично розташованими відносно площини екватора сфери і віддаленими від екватора кожна на 90 мм.

Внутрішня поверхня:

- правильна шестигранна призма з основою в горизонтальній площині проекцій і висотою 180 мм, центр основи призми співпадає з центром нижньої основи зрізаної сфери;
- пряма п'ятигранна призма з основою у площині, паралельній фронтальній площині проекцій і такою висотою, що призма повністю перетинає зовнішню поверхню сфери.

Другий етап розв'язання задачі - побудова тривимірних моделей окремих геометричних форм за допомогою команд **AutoCAD**.

Кожний геометричний елемент створюється окремо з урахуванням розмірів форми і положення. Способів для отримання таких моделей в **AutoCAD** існує декілька. Наприклад, зрізану сферу можна побудувати двома способами.

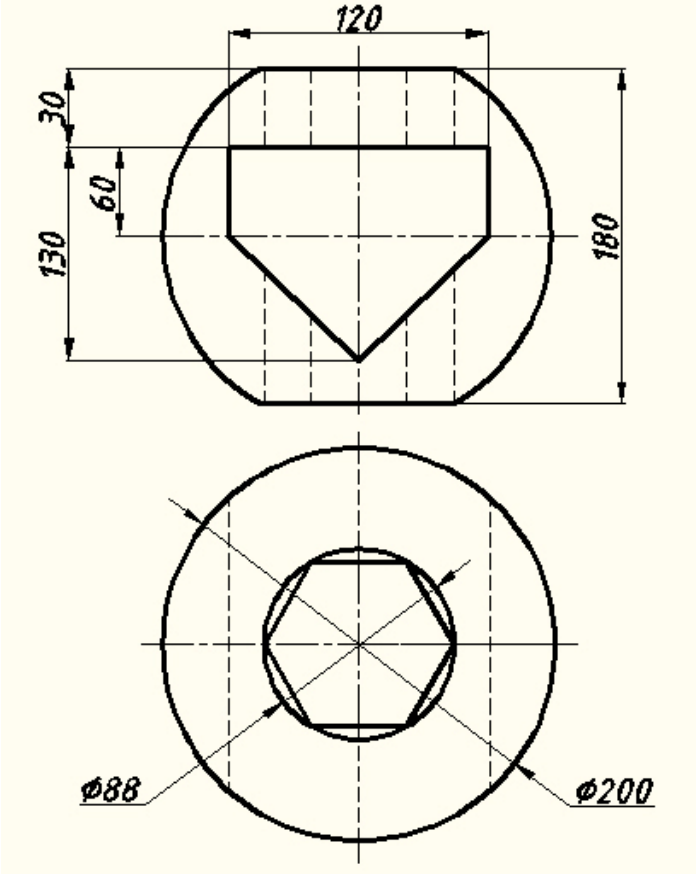


Рис. 1. **Завдання геометричної моделі**

I спосіб – побудова повної сфери за допомогою команди **Sphere** (меню **Solid** підменю **Priminive**)) з наступним зрізанням її площинами командою **Slice** (меню **Solid** підменю **Solid edition**).

II спосіб – побудова зрізаної сфери командою обертання **Revolve** (меню **Solid** підменю **Solid**) замкненого контуру навколо осі.

Твердотільну модель геометричного елементу, що утворює внутрішню поверхню вертикального отвору – правильної шестигранної призми, можна отримати за допомогою команди виштовхування плоского контуру **Extrude** (меню **Solid** підменю **Solid**).

Модель геометричного елементу, що утворює внутрішню поверхню поперечного отвору – п'ятигранної призми будується аналогічно, але з урахуванням того, що основа призми належить фронтальній площині.

Напрямок виштовхування перпендикулярний до площини контуру виштовхування чи задається шляхом. Отже, основу призми необхідно будувати після обертання системи координат навколо осі X чи Y на 90^{o} за допомогою команди **UCS** опція **X** чи **Y** відповідно (див. рис. 2).

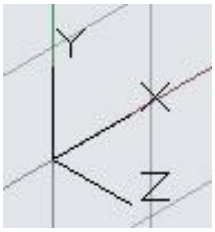


Рис. 2. **Система координат після обертання навколо осі X**

В результаті попередніх дій отримано моделі трьох тіл, що задають зовнішню і внутрішню поверхні тіла складної форми (див. рис. 3).

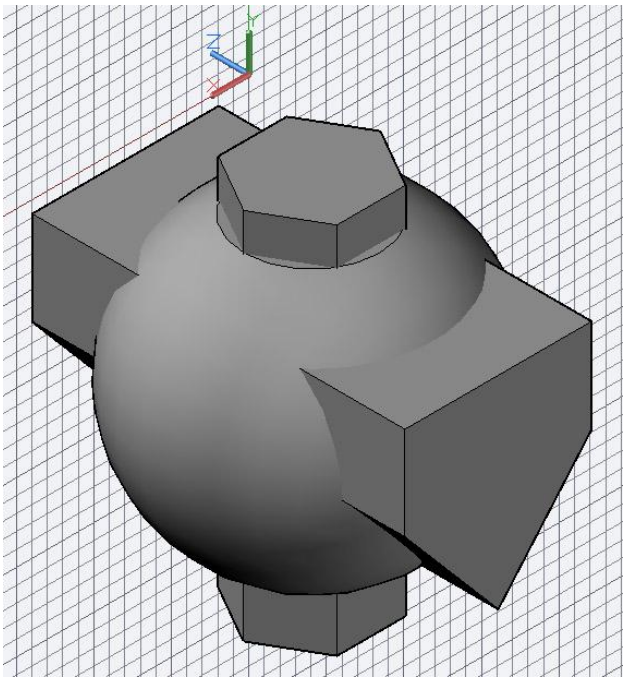


Рис. 3. **3D-моделі формоутворюючих геометричних елементів**

Для досягнення кінцевого результату необхідно виконати операцію віднімання тіл, що задають внутрішню форму, від тіла, що задає зовнішню форму моделі. Тобто, призми відняти від сфери. Для рішення цієї задачі використовується команда **Subtract** (меню **Solid** підменю **Boolean**). На цьому просторове розв'язання задачі завершено (Рис. 4).

Останній етап розв'язання поставленої задачі - отримання компоновки кресленика тіла складної форми за наявною тривимірною моделлю. Для цього використовується команда **Solview** (меню **Home** підменю **Modeling** команда **Solid View**) з попереднім створенням чистого нового аркуша і системою координат користувача, що відповідає головному виду кресленика.

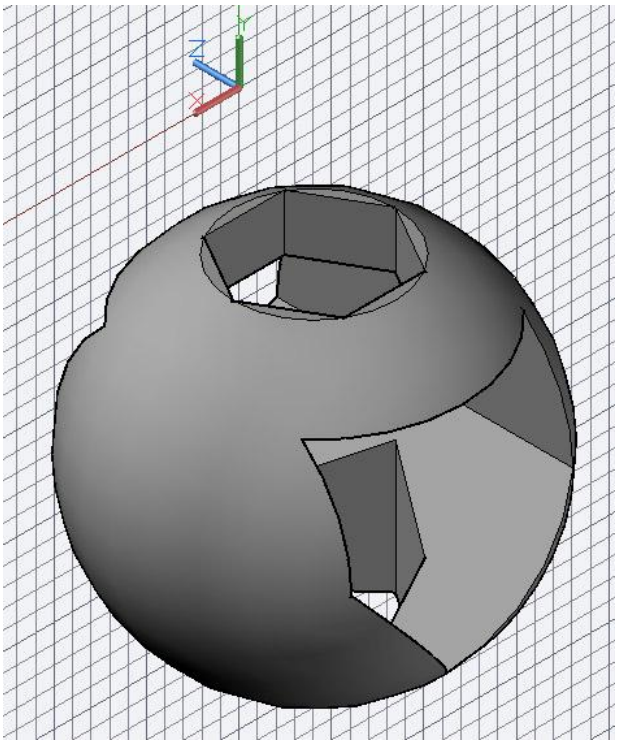


Рис. 4. **3D-модель заданого тіла складної форми**

Для отримання головного виду і початку компонування використовується опція команди **Solview - Ucs** (система координат користувача), для отримання інших потрібних видів у проекційному зв'язку з головним – опція **Ortho,** для отримання корисних розрізів – опція **Section**. В результаті маємо компоновку шести незалежних видових екранів на аркуші (три екрани – види, три екрани – розрізи). На наступному кроці треба вирівняти зображенні в екранах, в яких містяться розрізи тіла з зображеннями в екранах з відповідними видами. Для цього використовується команда **Mvsetup**.

Побудова проекцій видів і розрізів з штрихуванням фігури перерізу у площині аркуша виконується командою **Soldraw** (меню **Home** підменю **Modeling** команда **Solid Draw**). В результаті отримуємо плоскі зображення - види і розрізи з штрихуванням, що за замовчанням використовується в **AutoCAD**.

Штрихування можна відредагувати в режимі **Model** за допомогою діалогового вікна **Hatch Edit**, яке викликається подвійним клацанням мишею на штрихуванні.

В результаті дії команди отримання компоновки **Solview** видимі і невидимі лінії для кожного екрана розподіляються за окремими автоматично створеними шарами, імена яких містять інформацію про призначення і видимість шару (з частинкою **VIS** - видимі, **HID** - невидимі лінії, **HAT** – штрихування, **DIM** - розміри). Після вимкнення шарів з невидимими на екранах лініями та шару з межами видових екранів (**VPORTS)** на компоновці викреслюються осі симетрії та проставляються розміри, позначаються розрізи. Вона набуває вигляд, що показаний на рис. 5.

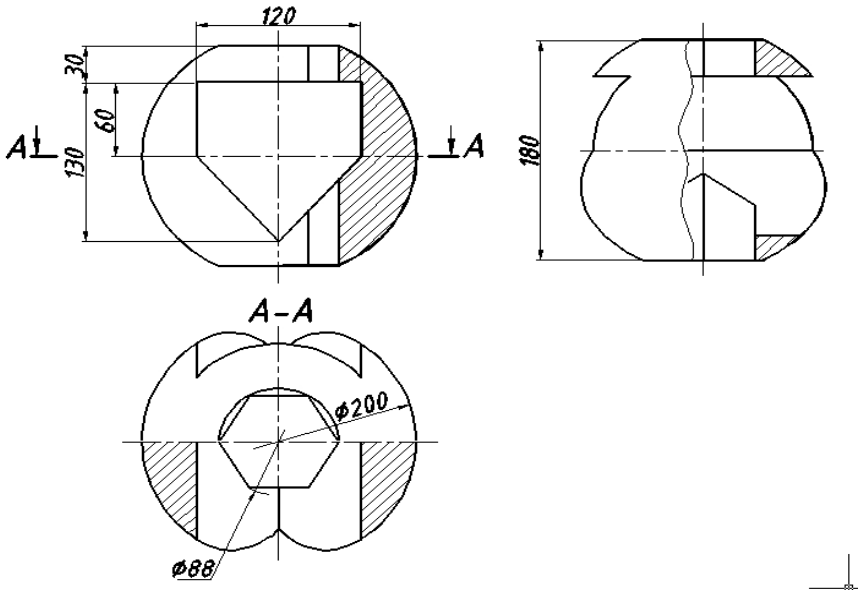


Рис. 5. **Компоновка кресленика тіла складної форми**

Висновки. Запропоновано основні етапи побудови тривимірної моделі та компоновки кресленика тіла складної форми в середовищі **AutoCAD 2015.**

1. Ванін В.В. Комп'ютерна інженерна графіка в середовищі AutoCAD : Підручник / В.В. Ванін, В.В. Перевертун, Т.М. Надкернична – К.: Каравела, 2006.- 336с.
2. Ванін В.В. Інженерна графіка: Підручник / В.В. Ванін, В.В. Перевертун, Т.М. Надкернична, Г.Г. Власюк. – К.: BHV, 2009.- 400с.
3. Бакалова В.М. Особливості моделювання геометричних образів на прикладі алгоритмічного розв'язання задач / В.М. Бакалова, Г.В. Баскова // Міжвідомчий науково-технічний збірник «Прикладна геометрія та інженерна графіка». Вип.88.- К.:КНУБА, 2011р.- с.66-71.
4. Бакалова В.М. Алгоритм розв'язання інженерних задач графічним та аналітичним способом / Г.В. Баскова, Л.Г. Овсієнко // Збірник праць XIV міжнародної науково-практичної конференції "Сучасні проблеми геометричного моделювання". –Мелітополь: ТДАТУ, 2012, с.3-10.
5. Михайленко В.Є. Інженерна та комп'ютерна графіка: Підручник.-6-те вид. / В.В. Ванін, С.М. Ковальов.- К.: Каравела, 2012.- 368с

УДК 515.2:563.3
Бакалова В.М., Баскова Г.В., Яблонський П.М.
Національний технічний університет України«Київський політехнічний інститут»

ВИКОРИСТАННЯ МЕТОДУ ПЕРПЕНДИКУЛЯРА ДОПРЯМОЇ ПРИ РОЗВ'ЯЗАННІ ІНЖЕНЕРНИХ ЗАДАЧ

Бакалова В.М., Баскова Г.В., Яблонський П.М. Використання методу перпендикуляра до прямої при розв'язанні інженерних задач.В статті розглядається оригінальний метод використання перпендикуляра до прямої при розв'язанні інженерних та складних геометричних задач.
Ключові слова:інженерні задачі, алгоритм, моделювання, обчислення, графоаналітичні способи.

Бакалова В.Н., Баскова Г.В., Яблонский П.Н. Использование метода перпендикуляра к прямой при решении инженерных задач.В статье рассмотрен оригинальный метод использования перпендикуляра к прямой при решении инженерных и сложных геометрических задач.
Ключевые слова: инженерные задачи, алгоритм, моделирование, вычисления, графоаналитические способы.

Bakalova V., Baskova G., Yablonskiy P. Some aspectsof graphanalytical method for solving engineeringproblems by the methodofperpendicular to the line. The article describesan original methodof usingperpendicular to the linesin the solutionof engineeringand complexgeometric problems.
Keywords: engineering problems, algorithm, simulation,calculation, graphic&analyticalmethods.

Постановка проблеми.Динаміка сучасних ринкових процесів висуває нові вимоги до якості професійної підготовки фахівців інженерно-технічного профілю. В умовах імплементації Закону України «Про вищу освіту» особливого значення набувають нові форми і методи навчання технічних дисциплін, які вимагають сучасного підходу і відповідного методичного забезпечення [1, 4].

Аналіз останніх досліджены публікацій.Якість технічної освітистудентіві їх конкурентоспроможність, як майбутніх фахівців-інженерів, суттєво залежить від рівня підготовки за такими базовими дисциплінами як геометрія, стереометрія, нарисна геометрія, креслення тощо [2]. Набуття знань та навичок при вивченні цих дисциплін єосновоюпри вирішенні прикладних інженерних задач.

Невирішені частини проблеми.Отже, розв'язання студентами інженерних задач при вивченні курсу «Нарисна геометрія та інженерна графіка» повинне формувати вміння не лише застосовувати класичні методи розв'язку, а спонукати до аналізу ефективності використання того чи іншого методу при вирішенні конкретної поставленої задачі.

Метою дослідження єпідвищення якості та ефективності навчального процесу студентів.Розглянемодеякі аспектиграфоаналітичних способів розв'язання інженерних задач.

Основнірезультати дослідження.Просторове, логічне та алгоритмічне мислення студентів досягається завдяки розв'язанню метричних та позиційних задач з нарисної геометрії[2, 3]. Розглянемо декілька способів побудови зображення перпендикуляра до прямої.

Устереометрії,маючи справу ззображеннямифігур,актуальною є задачапобудови зображенняперпендикуляраопущеного зточкинапряму, що дозволяє вирішити цілийряд завдань. Розглянемо розв'язання цієї задачі на прикладі, який наведений на рис.1.

НаребріABкубазаданаточкаP-серединацьогоребра.Побудуватипряму, що проходитьчерез точкуPперпендикулярнопрямій B_1D.

Поставлену задачу розв'яжемо декількома способами:

1. Перший спосіб – спосіб виносних креслеників (див.рис.2).

З'єднаємо точку P з точками B_2 і D. Побудуємо трикутник подібний оригіналу B_2DP.

1.1. Фігурою подібної оригіналу грані ABCD є квадрат $A_0B_0C_0D_0$. Відрізок D_0P_0 - один із сторін трикутника B_2DP.

1.2. Фігурою подібної оригіналу трикутника - квадрат AA_1B_1B. Відрізок $P_0(B_1)_0$ - друга сторона трикутника.

1.3. Аналогічно - третя сторона $(B_1)_0D_0$.

1.4. У трикутнику $P_1(B_1)_0D_0$ будуємо перпендикуляр $P_0H_0 \perp (B_1)_0D_0$.

1.5. За допомогою променя l, проведеного через точку B_1 будуємо точку H таку, що $B_1H: B_1D=(B_1)_0H_0: (B_1)_0D_0$.

1.6. Будуємо пряму PH.

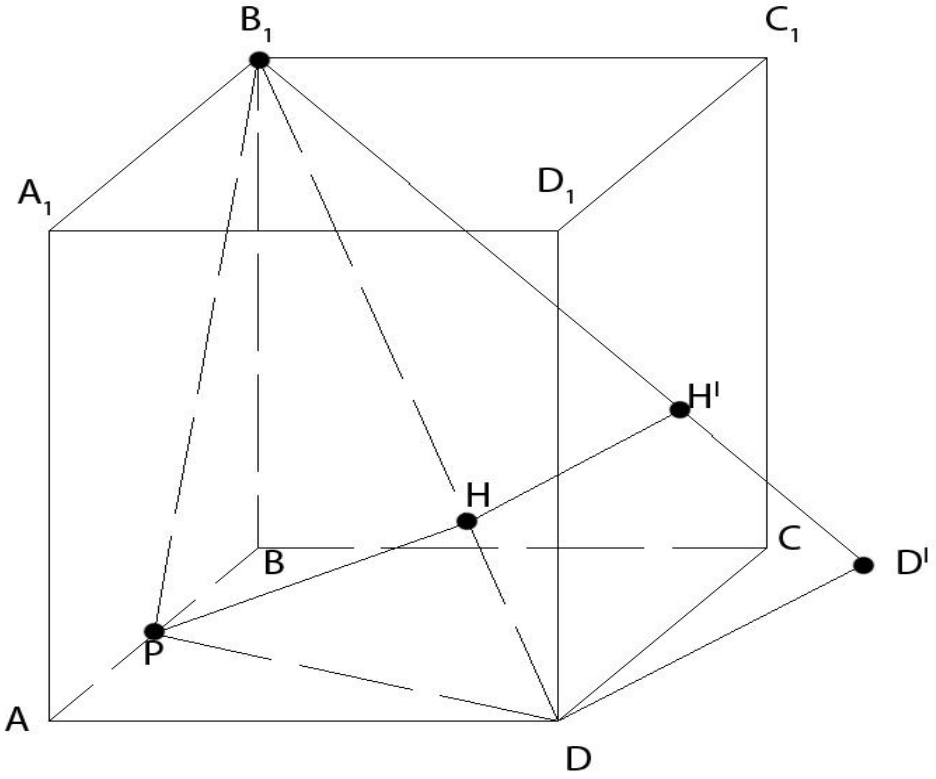


*Рис.1.***Графічна умова завдання**

2. Другий спосіб – обчислювальний.

2.1. Обчислимо сторони трикутника PB_1D за умовою, що сторона куба рівна a.

$$DP = \sqrt{AD^2 + AP^2} = \frac{a\sqrt{5}}{2}; \quad (1)$$

$$PB_1 = PD; \quad (2)$$

$$B_1D = \sqrt{BB_1^2 + BD^2} = a\sqrt{3} \quad (3)$$

2.2. $PB_1^2 - B_1H^2 = PD^2 - DH^2$ (4)

або $\frac{5a^2}{4} - B_1H^2 = \frac{5a^2}{4} - \left(a\sqrt{3} - B_2H\right)^2$; (5)

$$B_1H = \frac{3a}{2\sqrt{3}}. \quad (6)$$

Тоді: $B_1H : B_1D = \frac{3a}{2\sqrt{3}} : a\sqrt{3} = 1:2$ (7)

2.3. За допомогою допоміжного променя l будуємо на стороні B_1D точку H, таку, що $B_1H : B_1D = 1:2$

2.4. Будуємо пряму PH.

3. Третій спосіб – геометричний.

Якщо прямокутні трикутники ADP и BB_1P рівні (за двома катетами), то $DP = B_1P$. Це означає, що PH цього трикутника є і його висотою, тобто пряма PH є шуканою прямою.

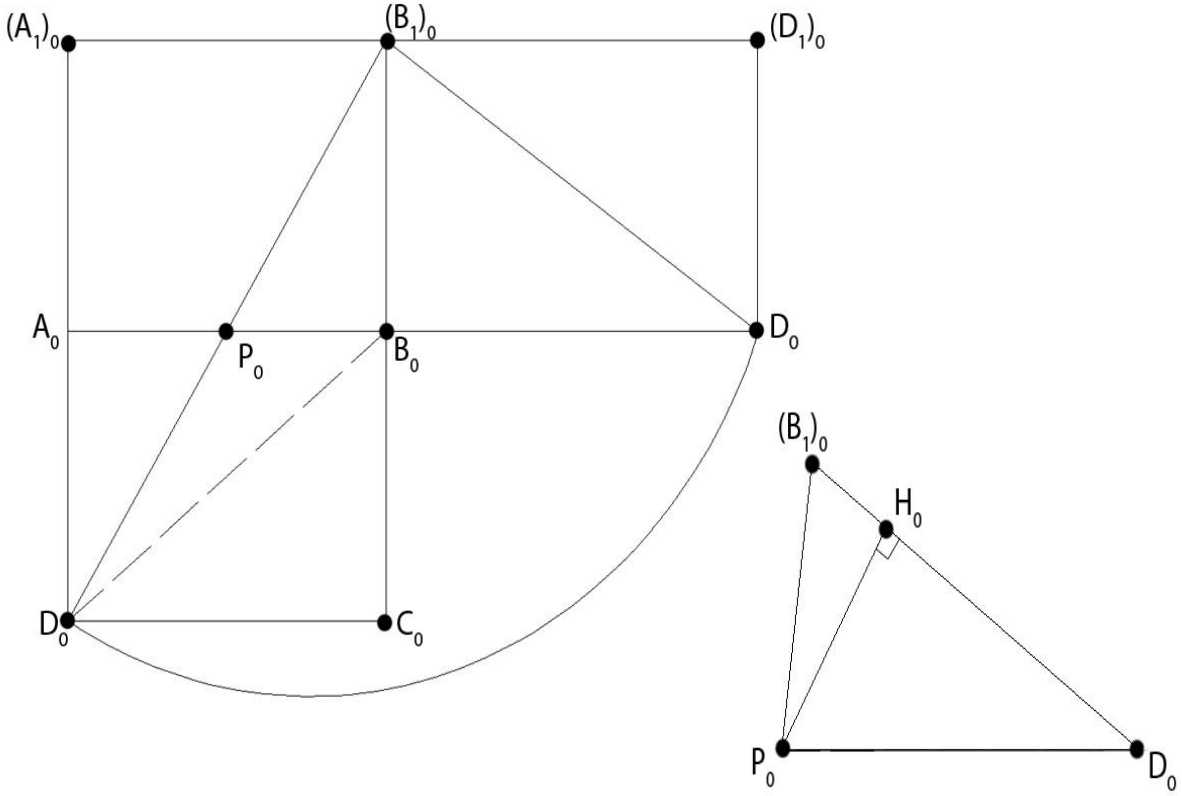


*Рис.2.***Розв'язання задачі способом виносних креслеників**

4. Метод прямокутного проекціювання на дві взаємноперпендикулярні площини (див. рис.3).

Алгоритм:

1. $B_1D = B_1 \cup D$
2. $P \subset \Sigma(\mathbf{f} \cap \mathbf{h}) \perp B_1D$
3. $H = B_1D \cap \Sigma$
4. $PH = P \cup H$

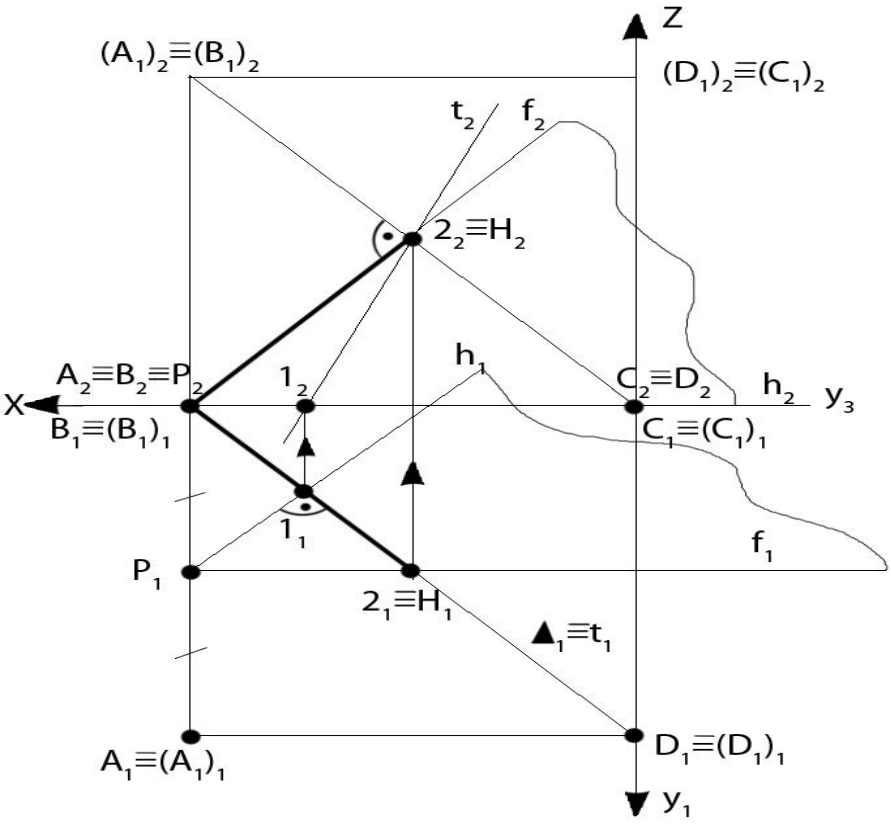


Рис.3

При розгляді більш складних фігур:призм, пірамід та інших, перші три способи розв'язання задач ускладнюються.Найбільш ефективними є способи нарисної геометрії, а при необхідності математичного моделювання, використання рівнянь аналітичної геометрії.

Висновки.З метою розвитку просторового уявлення студентів використано одну з найпоширеніших задач геометрії і застосовано багатосторонній підхід при її вирішенні. Отримана одна із багатьох відповідей на запитання, для чого вивчаються курси нарисної та аналітичної геометрії в процесі підготовки інженерів як творчої професії.

1. Ванін В.В. Інженерна графіка: Підручник / В.В. Ванін, В.В. Перевертун, Т.М. Надкернична, Г.Г. Власюк. – К.: BHV, 2009.- 400с.
2. Бакалова В.М. Особливості моделювання геометричних образів на прикладі алгоритмічного розв'язання задач / В.М. Бакалова, Г.В. Баскова // Міжвідомчий науково-технічний збірник «Прикладна геометрія та інженерна графіка». Вип.88.- К.:КНУБА, 2011р.- с.66-71.
3. Бакалова В.М. Алгоритм розв'язання інженерних задач графічним та аналітичним способом / Г.В. Баскова, Л.Г. Овсієнко // Збірник праць XIV міжнародної науково-практичної конференції "Сучасні проблеми геометричного моделювання". –Мелітополь: ТДАТУ, 2012, с.3-10.
4. Михайленко В.Є. Інженерна та комп'ютерна графіка: Підручник / В.В. Ванін, С.М. Ковальов.- К.: Каравела, 2004.- 344с.

УДК 515.2:514.18
Верещага В.М. д.т.н., Конопацький Є.В. к.т.н., Павленко О.М. аспірант
Мелітопольський державний педагогічний університет імені Богдана Хмельницького

ВИЗНАЧЕННЯ ПЛОЩІ, ОБМЕЖЕНОЇ ТОПОГРАФІЧНОЮ ЗАМКНЕНОЮ ПЛОСКОЮ КРИВОЮ

Верещага В.М., Конопацький Є.В., Павленко О.М. Визначення площі, обмеженої топографічною замкненою плоскою кривою. Запропоновано спосіб визначення площі, обмеженої довільною топографічною замкненою кривою, шляхом її поділення на окремі сегменти. Подаючи отримані сегменти дуги у вигляді окремих точкових рівнянь, обчислюються площі обмежені цими сегментами. Шукана площа знаходиться як сума площ під сегментами.

Верещага В.М., Конопацкий Е.В., Павленко А.М. Определение площади, ограниченной топографической замкнутой плоской кривой. Предложен способ определения площади, ограниченной произвольной топографической замкнутой кривой, путем её разделения на отдельные сегменты. Подавая полученные сегменты дуги в виде отдельных точечных уравнений, вычисляются площади ограниченные этими сегментами. Искомая площадь находится как сумма площадей под сегментами.

Vereschaga V.M., Konopatskiy E.V., Pavlenko A.M. Determining the area, topographic limitations locked plane curves. A method for determining the area bounded by topographic arbitrary closed curve by its division into separate segments. By submitting obtained arc segments as individual point equations are calculated area limited by these segments. The required area is the sum of the areas under segments.
Ключові слова: замкнена крива, сегменти кривої, точкові рівняння, площа.

Постановка проблеми. Задачі, у яких виникає необхідність визначення площі, обмеженої дугою кривої, виникають доволі часто. У разі, якщо крива задана рівнянням, застосовується її інтегрування. У разі, якщо вона дискретно представлена, застосовується дискретне інтегрування [1]. У разі, якщо дуга кривої представлена точковим рівнянням [2] – виникає проблема, яку треба усунути.

Формування мети статті. Розробити спосіб визначення площі сегмента, обмеженого дугою кривої, що задана точковим рівнянням.

Основна частина. Спочатку розглянемо площу сегменту обмеженого дугою цієї кривої.

Нехай у симплексі *CAB* задана точковим рівнянням дуга кривої (рис. 1):

$$M=(A-C)p+(B-C)q+C, \qquad (1)$$

де – p і q параметри уздовж *CA* і *CB*, відповідно, які визначаються простим відношенням трьох точок.

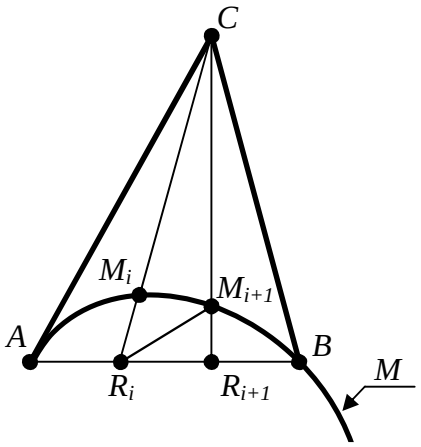


Рис. 1. **Схема для визначення площі між відрізком АВ і дугою АВ кривої М.**

Обираємо дві довільні точки M_i *і* M_{i+1}. Треба зауважити, що чим менша відстань між ними, тим точніше буде обчислена шукана площа сегменту. Запишемо точкові рівняння цих точок, що визначають їх положення на кривій *M*:

$$M_i = (A - C)\, p_i + (B - C) q_i + C, \text{ де } p_i = p(t_i),\ q = q(t_i); \tag{2}$$

$$M_{i+1} = (A - C) p_{i+1} + (B - C) q_{i+1} + c, \text{ де } p_{i+1} = p_{i+1}\ (t_{i+1}), q_{i+1} = q_{i+1}(t_{i+1}). \tag{3}$$

Із простого відношення трьох точок $M_i CR_1$ визначимо R_i: $M_i CR_i = r_i$. Якщо просте відношення подати у геометричній формі як відношення відрізків, то отримаємо:

$$r_i = \frac{M_i R_i}{CR_i},$$

а потім перейдемо до точкової форми простого відношення трьох точок прямої, тоді дістанемо:

$$\frac{M_i - R_i}{C - R_i} = r_i.$$

Якщо отримане відношення розв'язати відносно R_i, то

$$M_i - R_i = Cr_i - R_i r_i,$$

звідси

$$R_i(1 - r_i) = M_i - C_i,$$

відповідно:

$$R_i = \frac{M_i - Cr_i}{1 - r_i}, \text{ де } r_i = 1 - p_i - q_i. \tag{4}$$

Підставляємо в (4) точкове рівняння (2), отримаємо:

$$R_i = \frac{(A-C)p_i + (B-C)q_i + C(1-r_i)}{1-r_i} = \frac{(A-C)p_i + (B-C)q_i + C(1-1+p_i+q_i)}{1-1+p_i+q_i},$$

звідки маємо:

$$R_i = (A - C)\frac{p_i}{p_i + q_i} + (B - C)\frac{q_i}{p_i + q_i} + C. \tag{5}$$

Аналогічним чином, для простого відношення $M_{i+1}CR_{i+1}$ визначимо R_{i+1} $M_{i+1}CR_{i+1} = r_{i+1}$, якому віповідає геометрична форма у вигляді відношення відрізків:

$$\frac{M_{i+1}R_{i+1}}{CR_{i+1}} = r_{i+1}.$$

Від цього відношення маємо змогу перейти до точкової форми простого відношення трьох точок:

$$M_{i+1}CR_{i+1} \to \frac{M_{i+1} - R_{i+1}}{C - R_{i+1}} = r_{i+1}.$$

Розв'яжемо отримане відношення у точковій формі відносно R_{i+1}:

$$M_{i+1} - R_{i+1} = Cr_{i+1} - R_{i+1}\ r_{i+1} \to R_{i+1}(1 - r_{i+1}) = M_{i+1} - Cr_{i+1},$$

в результаті дістанемо:

$$R_{i+1} = \frac{M_{i+1} - Cr_{i+1}}{1 - r_{i+1}}, \tag{6}$$

Якщо у (6) підставити точкове рівняння (3) і виконати перетворення:

$$R_{i+1} = \frac{(A-C)p_{i+1} + (B-C)q_{i+1} + C(1-r_{i+1})}{1-r_{i+1}} = \frac{(A-C)p_{i+1} + (B-C)q_{i+1} + C(p_{i+1}+q_{i+1})}{p_{i+1}+q_{i+1}},$$

то у результаті отримаємо:

$$R_{i+1} = (A - C)\frac{p_{i+1}}{p_{i+1}+q_{i+1}} + (B - C)\frac{q_{i+1}}{p_{i+1}+q_{i+1}} + C. \tag{7}$$

Площа шуканого чотирикутника $S(M_iR_iR_{i+1}M_{i+1})$ (рис.1) дорівнює сумі площ двох трикутників:

$$S_{i,i+1}(M_iR_iR_{i+1}M_{i+1}) = S(M_iR_iM_{i+1}) + S(M_{i+1}R_iR_{i+1}).$$

На основі S-теореми [2], площа $S(M_iR_iM_{i+1})$ визначається добутком площі ΔABC, що дорівнює $S_{ABC} = \frac{ab\sin\gamma}{2}$, на визначник, рядки якого є параметрами точкових рівнянь (2), (5), (3), тобто:

$$\begin{vmatrix} p_i & q_i & 1 \\ p_i & q_i & p_i+q_i \\ p_{i+1} & q_{i+1} & 1 \end{vmatrix},$$

тоді площу $\Delta S(M_iR_iM_{i+1}) = \Delta S_i$ можна записати наступним чином:

$$\Delta S_i = \frac{ab\sin\gamma}{2(p_i+q_i)}\begin{vmatrix} p_i & q_i & 1 \\ p_i & q_i & p_i+q_i \\ p_{i+1} & q_{i+1} & 1 \end{vmatrix}.$$

Аналогічним чином складається визначник для площі $S(M_{i+1}R_iR_{i+1})$, з використанням точкових рівнянь (3), (5), (7). На основі тієї ж *S*-теореми площа трикутника $S(M_iR_iM_{i+1}) = S_{i+1}$ буде визначатися наступним чином:

$$\Delta S_{i+1} = \frac{ab\sin\gamma}{2(p_i+q_i)(p_{i+1}+q_{i+1})}\begin{vmatrix} p_i & q_i & 1 \\ p_i & q_i & p_i+q_i \\ p_{i+1} & q_{i+1} & 1 \end{vmatrix}.$$

Тут в ΔS_i та ΔS_{i+1} треба розуміти, що кут γ - це кут при вершині симплекса *C*. Тоді площу чотирикутника $S(M_{i+1}R_iR_{i+1}) = S_{i,i+1}$ можна добути з виразу:

$$S_{i,i+1} = \frac{ab\sin\gamma}{2(p_i+q_i)}\left(\begin{vmatrix} p_i & q_i & 1 \\ p_i & q_i & p_i+q_i \\ p_{i+1} & q_{i+1} & 1 \end{vmatrix} + \frac{1}{p_{i+1}+q_{i+1}}\begin{vmatrix} p_{i+1} & q_{i+1} & 1 \\ p_i & q_i & p_i+q_i \\ p_{i+1} & q_{i+1} & p_{i+1}+q_{i+1} \end{vmatrix}\right) \tag{8}$$

Враховуючи з (4), що $r_i = 1 - p_i - q_i$ розкриємо визначники з (8), в результаті отримаємо:

$$\begin{vmatrix} p_i & q_i & 1 \\ p_i & q_i & p_i+q_i \\ p_{i+1} & q_{i+1} & 1 \end{vmatrix} = \begin{vmatrix} p_i & q_i & 1 \\ 0 & 0 & -r_i \\ p_{i+1} & q_{i+1} & 1 \end{vmatrix} = -r_i(p_iq_{i+1} - q_ip_{i+1}), \tag{9}$$

У цьому перетворенні визначника від його другого рядка було віднято перший та результат записано до другого рядка перетвореного визначника:

$$\begin{vmatrix} p_i & q_i & 1 \\ p_i - p_i & q_i - q_i & -1 + p_i + q_i \\ p_{i+1} & q_{i+1} & 1 \end{vmatrix}.$$

$$\begin{vmatrix} p_{i+1} & q_{i+1} & 1 \\ p_i & q_i & p_i+q_i \\ p_{i+1} & q_{i+1} & p_{i+1}+q_{i+1} \end{vmatrix} = \begin{vmatrix} p_{i+1} & q_{i+1} & 1 \\ p_i & q_i & p_i+q_i \\ 0 & 0 & -r_{i+1} \end{vmatrix} = -\, r_{i+1}(p_i q_{i+1} - p_{i+1} q_i) \qquad (10)$$

Якщо для (8), (9) та (10) ввести позначення $\Delta_{i,i+1} = p_i q_{i+1} - p_{i+1} q_i$, то тоді отримаємо формулу для обчислення площі чотирикутника $M_i R_i R_{i+1} M_{i+1}$, сторона якого $M_i M_{i+1}$ є дугою кривої *M* (рис.2).

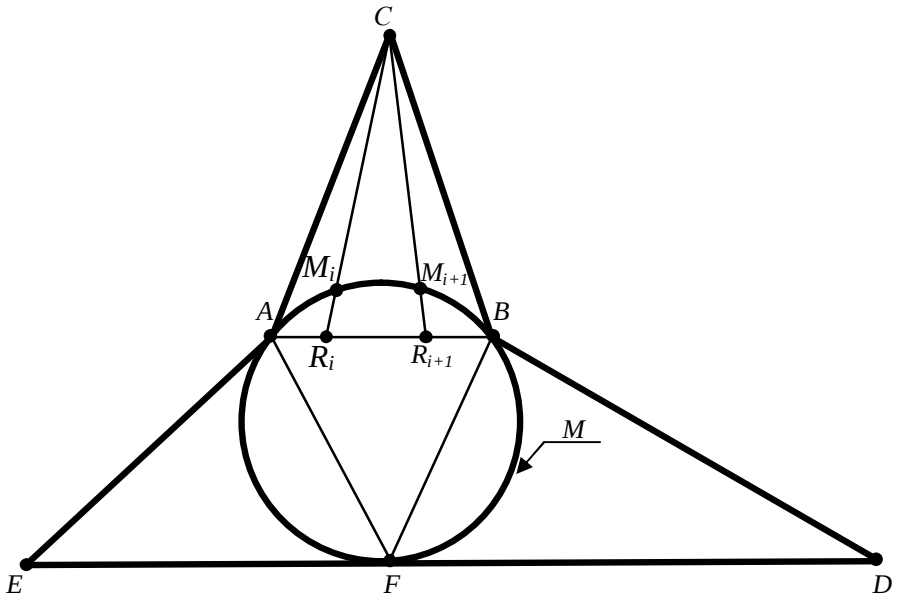


Рис. 2. **Геометрична схема для замкненої кривої М**

Для підвищення точності розрахунків площ необхідно зменшити відстань між точками M_i та M_{i+1}. Тепер треба обчислити площу S_c, що знаходиться між дугою AB та відрізком прямої AB. У відповідності до рис.1 вона буде дорівнювати сумі:

$$S_{i,i+1} = \frac{ab\sin\gamma}{2}\left(\frac{r_{i+1} - r_i(p_{i+1}+q_{i+1})}{(p_i+q_i)(p_{i+1}+q_{i+1})}\Delta_{i,i+1}\right) \qquad (11)$$

$$S_c = S_{i,i+1} + S(AR_iM_i) + S(M_{i+1}R_{i+1}B) \qquad (12)$$

Треба зауважити, що для іншої геометричної схеми права частина з (12) буде мати іншу кількість та якість доданків.

Дотепер було розглянуто визначення площі для дуги AB кривої *M*, заданої точковим рівнянням (1) відповідно до рис.1 для визначення площі S_M, обмеженої замкненою кривою *M*, у відповідності до рис.2, необхідно розглянути ще два симплекси *DBF* та *EFA* і повторити для них алгоритм, аналогічний алгоритму симплекса *CAB*. Тоді площа S_M може бути обчислена:

$$S_M = S_C + S_D + S_E + S_{\Delta ABF} \qquad (13)$$

Розв'язання для іншої геометричної схеми (13) буде мати відповідний вигляд.

Висновки. Розв'язання задачі визначення площі, обмеженої топографічною замкненою плоскою кривою, що представлена у вигляді точкового рівняння, дозволить обчислювати об'єми горбів та западин на рельєфах земельних ділянок.

1. Верещага В.М. Дискретно-параметрический метод геометрического моделирования кривых линий и поверхностей. Дис. докт.техн.наук: 05.01.01 – Мелитополь, 1996. – 320с.
2. Балюба И.Г. Конструктивная геометрия многообразий в точечном исчислении. Дис. докт.техн.наук: 05.01.01 – Макеевка, 1995. – 227с.
3. Конопацький Є.В Геометричні моделювання алгебраїчних кривих та їх використання при конструюванні поверхонь у точковому численні Балюби-Найдиша: канд. техн. наук. – Мелітополь: ТДАТУ, 2012 – 26с.
4. Кучеренко В.В. Формалізовані геометричні моделі нерегулярної поверхні для гіперкількісної дискретної скінченої множини точок: Мелітополь: ТДАТУ, 2013 – 232с.

УДК 013.77:004.42; 37.013.03:004. 588(073)
Holovin M., Golovina N.
Eastern European National University of LesyaUkrainka, Lutsk,Ukraine

PROCESS OF PROGRAMMING STUDYING PROCESS IN THE CONTEXT OF LIMITED ATTENTION FIELD

Holovin M., Golovina N. Process of programming studying process in the context of limited attention field. The processes of programming studying in the context of peculiarities of thinking were researched. These peculiarities are conditioned by the limitation of the attention field, and structure of studying material. Attention field and consciousness are connected by the short – term memory. Three component memory model and conception about evaluation of memory field size, through the magic number of Miller were used for explanation of studying processes. The explanation of three component memory model through peculiarities of brain structure is presented. короткочасної пам'яті. In particular cyclical stimulation of limbic system was regarded as an embodiment of short - term memory. The localization of long-term memory traces and their hierarchical organization are also researched. This approach gave the possibility to connect in one model peculiarities of brain functioning, consciousness phenomenon, conceptions of education and emotional sphere of student. Original part of the research is related to graphical formalization of cognitive hierarchical scheme of long – term memory. Evolution of formation of this scheme is regarded in the article through the prism of educational strategies from general to concrete and from concrete to general.
Keywords: methodology of informatics studying, Miller magic number, attention, three component memory model, short-term memory, long-term memory, knowledge structure , studying of practical programming.

Головін М.Б., Головіна Н.М. Процес навчання програмуванню в контексті обмеженого поля уваги. Досліджувалися процеси навчання програмування в контексті особливостей мислення людини. Ці особливості обумовлені обмеженістю її поля уваги, а також структурою її знань. Поле уваги і свідомість пов'язані з короткочасною пам'яттю. Для пояснення процесів навчання в роботі використовуються трьохкомпонентна модель пам'яті і концепція оцінки розміру короткочасної пам'яті, через магічне число Міллера. Представлено пояснення механізму трикомпонентної моделі пам'яті через особливості будови мозку. Зокрема, циклічне збудження в лімбічної системи мозку розглядалося, як втілення короткочасної пам'яті. Розглядалося також ієрархічна специфіка організації довготривалої декларативної пам'яті. Цей підхід дав можливість зв'язати в одній моделі особливості функціонування мозку, феномен свідомості і поля уваги, концепції навчання, структуру знань, емоційну сферу учня. Оригінальна частина роботи стосується графічної формалізації когнітивної ієрархічної схеми в довгостроковій пам'яті. Еволюція формування цієї схеми розглядається в роботі через призму стратегій навчання від загального до конкретного і від конкретного до загального.
Ключові слова: методика інформатики, магічне число Міллера, увага, трьохкомпонентна модель пам'яті, короткочасна пам'ять, довготривала пам'ять, структура знань, вивчення практичного програмування.

Головин Н.Б., Головина Н.Н. Процесс обучения программированию в контексте ограниченного поля внимания Исследовались процессы обучения программированию в контексте особенностей мышления человека. Эти особенности обусловлены ограниченностью его поля внимания, а также структурой его знаний. Поле внимания и сознание связаны с кратковременной памятью. Для объяснения процессов обучения в работе используются трехкомпонентная модель памяти и концепция оценки размера кратковременной памяти, через магическое число Миллера. Представлено объяснение механизма трехкомпонентной модели памяти через особенности строения мозга. В частности, циклическое возбуждения в лимбической системе мозга рассматривалось, как воплощение кратковременной памяти. Рассматривалось также иерархическая специфика организации долговременной декларативной памяти. Этот подход дал возможность связать в одной модели особенности функционирования мозга, феномен сознания и поля внимания, концепции обучения, структуру знаний, эмоциональную сферу ученика. Оригинальная часть роботы касается графический формализации когнитивной иерархической схемы в долговременной памяти. Эволюция формирования этой схемы рассматривается в работе через призму стратегий обучения от общего к конкретному и от конкретного к общему.
Ключевые слова: методика информатики, магическое число Миллера, внимание, трехкомпонентная модель памяти, кратковременная память, долговременная память, структура знаний, изучение практического программирования.

Problem formulation

Attention field of a human has a limited amount of conceptual units, which he can simultaneously manipulate. This limitation puts a mark on the whole process of his thinking. Studying, as a basic part of intellectual activity process also has a specificity, which is connected with a limited field of attention. Let's examine the peculiarities of scientific activities in the context of these limitations.

The problem of developing new teaching methods and improving old is that pedagogy and teaching methods, as a science, don't study the human's brain and the mechanisms of cognitive processes. There is a problem of presenting the mechanism of attention switching, that interacts in the educational process with knowledge structure. Also, it's interesting to pay

attention to the structure of knowledge in the process of its formation. These mechanisms, presented in the generalized form, can give the basis for the improvement of methods of traditional and automatised learning. Cognitive psychology formed conceptual vision of cognitive processes, that can be succesfully used in various developments in pedagogy and methodology of education.

Analysis of last researches and publications

In the famous work of Miller (1956) "Magic number seven, plus or minus two" was shown that the amount of notions, which person can manipulate in the field of attention is 7±2 [1]. Process of remembering is conceptually well described by the three-component memory model [2]. In the cognitive psychology it's assumed that educational actions gradually form the reflection of educational object – its cognitive scheme, in the long - term declarative memory. This scheme is perceived only in parts. It's formed in the process of repeated multiple times attention switch. Each cognitive structure is a modification of the previous one [3, 4]. The cognitive structures evolutionize in the process of their differentiation [5], and as a result they are often forming hierarchicl structures.

The construction of teaching methods, which are based on mechanisms of cognitive psychology **is actual, but unresolved problem**. Cognitive psychology does not give direct recommendations concerning the learning process. There is a necessity to adopt the doctrines of psychology to the teaching methods. Programming is an especially convenient area of research in the context of the studying process. Here, the study material is strictly formalized, interconnected and structured. The structure of the material has the refined hierarchical type. The dynamics of the educational process can be represented as detailed of known information.

The aim of this work is to examine education through the prism of three-component learning memory model. On this basis the following binding must be realized: the functioning of the brain, the phenomenon of consciousness and fields of attention, learning concepts, knowledge structure, emotional sphere of the student. The objective of this work is also to formalize mental representations of the person, in the process of programming, in the form of graphic charts and analysis of the evolution of the formation of these cognitive circuits.

The main research

Three component memory model includes sensory, short-term and long term memory (Fig.1.). Short-term memory in this model is associated with the field of memory. Reasoning of studying processes is made, based on this model.

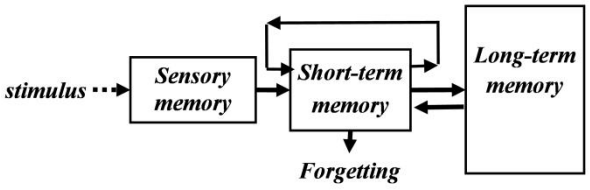


Fig.1 Three-component memory model

Process of remembering of information (studying) in the 3 component memory model is made in the following way. Information from the external environment gets firstly to the specific modal sensory registers of iconic (vision) and echoic memory (hearing), where it is stored for nearly a second in the form of physical stimulation trace. We will ignore the perception through the sensors like touch, smell and taste in the context of informatics. After going through sensory memory information is transferred to short-term storage place, with transcoding into verbally-acoustic form. If the information is not transcoded, because the person doesn't have relevant template for its classification and comparison with relevant words, than in several seconds the information is lost. For example student does not see in the text the program cycle, because he doesn't know how relevant operator looks like, and how the operator is structurally designed: does not know where starts and finishes its operation, doesn't imagine structural block-diagram of its functioning.

Remembering in the long-term memory on Fig 1. is represented by an arrow, which links short-term and long-term memory. Portion of new information easily goes through short-term memory and is

accumulated in the long term memory, if this portion of information fits into short-term memory and complements the structure of knowledge, which was created at the moment of recognition. For example the algorithm "sorting by searching for maximum" is easily understood, if before algorithm "searching for maximum" was assimilated. If conformity between new portion and already existing structure of knowledge is not reached, then new information can temporary stay in the short-term memory, by using verbal loop – cyclic spellings of the words (notions). Than the search of conformity between information in short-term and long term memory is made. If the conformity is reached than the portion of information is included into long-term memory.

In the short-term memory the attention and consciousness are incarnated. The material, which person recognizes, during the period of studying is always limited by the field of studies. Probably because of this in the language there are words of different stage of generalization and the possibility of scaling of notions in the process of studying abstract-logical intellectual actions. In this way the words "search for maximum" unfold during its program realization as a program fragment, that includes cycle, branching and checking of condition and two conferments. Even bigger block is associated with the word "sorting", because the first algorithm is a part of second.

Long-term memory accumulates information for an uncertain tim. Every portion of new material, that comes to long-term memory is transformed depending on the existing knowledge. Some part of information can quite precisely be saved during dozens of years, for example definitions, theorems, poems, etc. Part of information, which is used rarely "dissapeares" (goes to the subconsciousness). Long-term memory of a student is usually partly structured (organized) in some knowledge, and part of it represents fragments, which are different and not connected to each other. Those fragments can be accessible, but they are not making the whole picture. The process of studying is a process if differentiation of existing knowledge [5].

Problem definition

The narrowness of attention field puts a trace on the order of all studying processes. Development of programming techniques is straightly connected with specificity of abstract-logical thinking. Programming technologists empirically found method of downward step by step specification and module programming [6, 7] in which in refined shape the abstract-logical thinking is reflected and programming intellectual actions in the context of limited field of attention are optimized.

Three component memory model allows: to understand some important mechanisms for methodology of teaching, and basing on it to monitor field of attention and to support the process of solving of algorithmic tasks in context of the nearest perspective: one inductive step (generalizing) and one deductive step (specification). In the case of inductive action this activity should finish by editing of current program fragment and its checking. The amount of conceptual units, which students manipulate, doesn't have to overstate magic Miller number - 7. Complex of conceptual units should make logically finished construction.

In the context of studying practical programming the moment of excretion and recognition in the program body of program fragment, in which some logically finished amount of words is excreted and compared with single conceptual unit, which generalizes the activity of program fragment, that is considered.

In the context of studying programming the most actual are hierarchies and sequences. Sequences are reflected in the following: the program is a text, written sequentially from left to right and from top to bottom. From the other side this text has a logical connections, which are recognized as hierarchical construction. The process of practical-studying programming illustrates well the forming of hierarchical and sequential structure. It's understandable that all hierarchical construction of program (in all details), that is formed in declarative long-term memory and cannot be perceived at once: during the one session of memory concentration. Studying programs from 10 to more operators are meant. During several sessions of memory concentration , which are connected with each other it is possible to cover study program of several dozens of operators. The amount of professional programs is much bigger and can be comprehended only by transferring attention many times.

The original approach to understanding of programming learning

For example, on Fig. 2 very simple study program on Pascal language is shown. The program includes 16 operators in which sorting of letters in the line is made. Every operator is responsible for separate standard action in the program. Operators in the program are connected between each other by hierarchical superstructure, which is situated in the big triangle, which is marked by dotted line.

The circles, connected to each other mark conceptual constructions. In the toned triangles are shown the logically completed constructions, each of which can be covered during the one session of concentration of the field of attention. These triangles are named the constructs. It is seen that the creation of this program needs the solution of many intermediary tasks: the search of bigger between 2(if), exchange of indicators between two elements of the line (ob), single search of the maximum in the line (mx), sorting in the line (sr). The goal of the program is to realize text sorting in the file. To realize that it is necessary to solve extra tasks: downloading (zv) and saving of the file (zb).

Information, that is situated in the long term memory can be recognized only in small portions, the size of which is correlated with the size of short-term memory. The concentration of memory on one or other aspects of activity raises the attention in the short-term memory. This process can be named remembering.

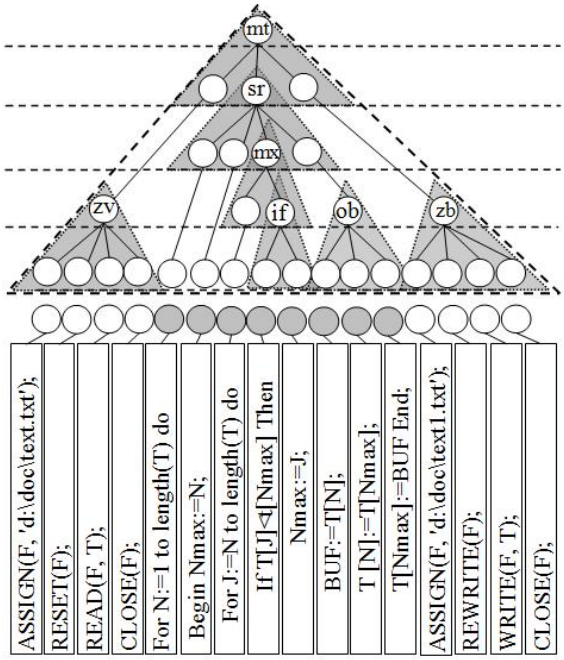


Fig.2 Simple study program and its cognitive structure

In relation to the presented above while studying task of programming it's possible to say the following. Integral mental reflection of the program, which covers all conceptual construction (the triangle is marked by dotted line) is formed in the long-term memory and is never realized fully in all details in the same time. Forming and awareness of the work mechanism is made in portions. Each portion can be covered by the field of attention. These portions are thought over in the short-term memory. It is understandable that thinking in the limits of construct mt is the thinking about the whole program in the most general treats and thinking in the limits of construct if or ob is maximally concretized.

Evolution of formation in the long-term declarative memory of the cognitive structure of the typical educational programming object, taking into account the attention are presented in the research [8].

On Fig.1 long term remembering is marked by the arrow from long term to long term memory. Memory is associated, because visual image can cause remembering of some word and vice versa.

Methodological aspect of teaching informatics in the context of views about short-term memory, needs the support of students during the process of solving of some additional tasks, which correspond to some constructs. Aspect of teaching is connected to long-term memory and needs methodological and diagnostic development, that support the creation in the declarative long term memory holistic hierarchical structures, which are the reflection of connections between separate constructions and connections between them. Moving between the constructs from up to down and from down to up are realized via refined abstract-logical thinking, that includes deductive and inductive activities, analysis, synthesis, abstraction, concretization and generalization. Effective forming of skills, abstract-logical,

causal intellectual activities in the process of learning informatics is very important and has big impact on general education and professional preparation. The hierarchy of cognitive structure of educational object is also mentioned in the research [9]. The last is important for the development of methodical approaches in education. The knowledge of structural organization of scientific material helps to optimize the process of education through the methodological means.

According to author views, represented model allows to explain uniformly – to homogenize big amount of memory phenomena, attention, appreciation and also pedagogical and methodological aspects of learning. Methodological consequences of work analyze of 3 component memory model in the sphere of informatics learning are especially interesting, because the represented model explains strict casualness of practical studying programming and abstract logical character of cognitive actions, and also this model gives the possibility to form conceptual methodological approaches to studying processes.

Neuro-cognitive proof to the study approach

Considering all spectrums of attractive moments there is one question, that is necessary to solve, before building methodological basement. This is a question about effectiveness of this model, how it responds neuro-cognitive peculiarities of brain structure. The reply to this question can be found in the works of Ivanitskij [10].

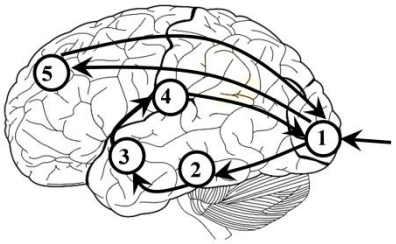


Fig.3 The consolidation of new information into long-term memory - associative cortex.
1.Projective visual cortex; 2.Associative temporal lobe; 3. Hippocamp; 4. Thalamus; 5 Frontal lobe

Short term memory in the mentioned works is regarded as circular motion excitation by system limbic system of brain, which includes projection and associative neocortex rind and some parts of middle brain . Short-term memory is not for long-term information saving. Loss of information from short-term memory is caused by fading excitation or displacement of one excitation by the other one, which is responsible for other field of attention.

Long-term declarative memory is connected with associated rind of medial sectors of temporal area of the cerebral hemispheres. Remembering in the long-term memory is realized via creation of neural connections in the associative brain rind. For realization of these connections there is a necessity of the long term excitation of one modality, which goes through appropriate temporal cortex area. The last is provided via many times repeating of information. Repeating of information several times renew neural connections and makes better the process of remembering.

Modally specific hearing and visual images are situated in associated secondary zones. In the tertiary zones the connection between modally specific images of different modality is realized. For example between a word and visual image.

For teaching it is very important to know that the components of the limbic system are responsible not only for short-term memory, attention and consciousness, but also for emotional state of person. It puts some limitations on teaching. Let's regard in more detail, in this context the processes of cyclic excitation, which is responsible for short-term memory.

Signals from sensation organs come to projection rind , then rind comes to associative rind – inferior temporal for visual stimuli (17,18 zones, hereinafter author mean Brodmann zones). There the information is compared to the standard and is recognized. Also the transcoding of information into verbal acoustic form 37.

Then excitation goes to entorial rind, which is situated on the internal part of the surface of the cortex temporal lobe (28). This rind plays the role of connector during the exchange of information between associated areas of neocortex and hippocampus.

Then impulses get to hippocampus. After that excitations move to motivation centers of diencephalon – thalamus and hypothalamus. Here emotional attitude to perceived information is formed. It's in hypothalamus the importance of signal for the needs of the organism is determined.

From there the excitations come back by the system of diffuse projections to the brain rind, including the zones of primary projections. The last instance is the same as primary. Excitation signal gets in these conditions the closed and cycled character. Thanks to return excited nerve impulses, that come from motivational centers of brain overlap in the projection rind on the traces of sensory excitation. This cycle, the duration of which is nearly 150 ms got the name "circle of feelings". In 100 ms after starting of the process of cycle excitation appear also the connections between projection and frontal cortex. Frontal cortex provides the control of activities and their planning [8].

Let's regard in more details the influence on emotional states of different parts of midbrain, in the context of studying processes (Fig.3).

Thalamus is responsible for the primary processing, transition and redistribution of information on the way from the sensors to the rind of cerebrum (smell is an exception). In thalamus there are 4 main cores, which are responsible for visual, auditory and tactile information and the feeling of harmony and balance. Thalamus plays also an important role in remembering. Also damaging of thalamus leads antiretrograd amnesia – moving of information from short-term to long-term memory.

Hypothalamus is situated under thalamus and defines the significance of signal for different needs of organism. Hypothalamus creates together with hypophysis a single functional complex, in which first one plays regulatory role and the other one effectory role. Hypothalamus supports organism in the limits of adaptive and homeostatic parameters, necessary for life support. It is responsible for thermoregulation, regulation of breathing, regulation of sleeping cycle, quenching of hunger, thirstiness and sexual attraction. Unbalance of these parameters is the strongest motivational and behavioral factor. Hypothalamus is also called "stress center". Its unbalance can destroy any studying process.

Amygdala is connected by nerve connections to hypothalamus and is responsible for the decision to attack or to run away, swallow or not and many other. Amygdala gives fast precognitive, affective evaluation of situation from the safety of life point of view. Amygdala takes part in forming negative (fear) and positive emotions (pleasure). That is why excitation, which are caused by aggression or fear influence studies a lot.

Hippocampus, "sea horse" is made up of 2 long structures, which are connected inside temporal lobes of the brain. Hippocampus allocates and keeps in the flow of external stimulus the important information, making a function of short-term memory manager. It takes part in coding the environment around us (spatial memory). In this way hippocampus transfer information to long-term memory. If it is damaged the syndrome of Korsakov appears, when a sick person loses short-term memory and keeps only long term memory.

Analysis of the model gives the following conceptual conclusions concerning the lessons.

Conclusion

1. Scalability concepts are the specific response on limitation of attention field during the thinking process. Abstract-logical thinking, as a phenomenon, is realized by various concepts with different ranges of generality. It is reflected on computer science seminars and lectures.

2. During the process of solving programming educational tasks, problem must be divided into logical subtasks, first, each of them are solved separately and, then the solutions are combined. Similarly, educational material on the lecture is divided into logical small parts. The size of new material in every part is not bigger than 7±2 (Miller`s number). Overloading of short term memory leads to the loss of certain elements from new material. Every part has intermediate conclusion. These conclusion combinations generate conclusions at the end of lecture.

3. Conceptual structure of the programming task solution, or the one, that is relevant to new lecture material should be hierarchically structured. Hierarchical structure means connected parts of solution or lecture. Moving in such hierarchical structure, during the lecture or during the process of solving problem is available in two ways: in strategy from general to concrete or from concrete to general.

4. The strategy of program creation from general to concrete requires step by step downward detailed algorithm. On every stage the size of part is not bigger than Miller`s number. In strategy from general to concrete, each new material part starts from conceptual position which is explained in mode of specification and application. At the end of the lecture or seminar information is finished by conclusions again.

5. Strategy from concrete to general on educational programming practice is embodied in the methodology of modular programming. This strategy from general to concrete creates general conclusion at the end of logical development and this conclusions can be logical ending of the lecture or seminar.

6. The formalization of the knowledge structure in a hierarchical structure opens the way to modeling representation of processes of thinking and learning. Mainly it concerns: software objects, classification of software and hardware computer technologies, databases, network Internet structures, schematic realization of the hardware equipment, software objects menu, etc.

7. In the process of teaching it is necessary to repeat several times new materials, which is good connected with previous material. For the creation of new neural connections the time is needed . In addition formation of new connections should be based on already formed neural structures. Any information can be completely new, each part is a modification of what we already have in our memory.

8. Hippocampus, amygdale, thalamus and hypothalamus are parts of limbic system and midbrain. They are responsible for the consolidation of new material into long-term memory, through short-term memory and for sensitive sphere. So information in short-term memory can be lost when you change your mood on the lecture. Attention focusing on abstract informatics themes cannot be achieved in case of: thirst, hunger, cold, fear, self-aggression, sexual arousal. However, low emotional background on the lecture creates a weak motivation to study new material and does not allow creating long circle excitations determination of modality to learn information in long-term memory.

1. Miller George A. The Magical Number Seven, Plus or Minus Two / George A. Miller // The Psychological Review. – 1956, – vol. 63. Issue 2. – P. 81-97.
2. Солсо Р. Когнитивная психология / Р. Солсо. — 6-е изд. — СПб.: Питер, 2006. — 589 с.
3. Холодная М. А. Психология интеллекта: парадоксы исследования / М. А. Холодная. – СПб. : Питер, 2002. – 272 с.
4. Найссер У. Познание и реальность. Смысл и принципы когнитивной психологии / У. Найссер. – М. : Прогресс, 1981. – 225 с.
5. Чуприкова Н.И. Психология умственного развития: Принцип дифференциации / Н.И. Чуприкова. –М.: Столетие, 1997. – 478 с.
6. Хьюз Дж. Структурный подход к программированию / Дж. Хьюз, Дж. Митчом. – М.: Мир, 1980. – 276 с.
7. Дал У. Структурное программирование / У. Дал, Э. Дейкстра, К. Хоар. – М.: Мир, 1975. – 246 с.
8. Головін М.Б. Зміст підготовки висококваліфікованого фахівця з інформаційних комп'ютерних технологій у контексті когнітивних процесів (на прикладі програмування) //Інформаційні технології в освіті. – Херсон, 2008.– Випуск 2. – С. 66-73.
9. Головін М.Б. Автоматизація тестування знань. Ієрархічні структури у комп'ютерних тестах / М.Б. Головін, О.І. Сомик // Інформаційні технології в освіті. – Херсон, 2011. – Випуск 10. – С. 058-063.
10. Иваницкий А.М. Сознание и мозг // В мире науки. – Москва, 2005. – N 11. – С. 3-11.

УДК 517:577.2:539.1:574.34
Губаль Г.М.
Луцький національний технічний університет

МАТЕМАТИЧНИЙ АНАЛІЗ ВПЛИВУ РАДІОАКТИВНОГО ОПРОМІНЮВАННЯ НА АУТОСОМНИЙ ГЕНОМ: МІЖДИСЦИПЛІНАРНІ ЗВ'ЯЗКИ

Губаль Г.М. Математичний аналіз впливу радіоактивного опромінювання на аутосомний геном: міждисциплінарні зв'язки. У статті зроблено математичний аналіз та виведено рівняння впливу радіоактивного опромінювання на аутосомний геном з використанням міждисциплінарних зв'язків.
Ключові слова: радіація, мутація, аутосомний геном, частота, різницеве рівняння.

Губаль Г.Н. Математический анализ влияния радиоактивного облучения на аутосомный геном: междисциплинарные связи. В статье выполнено математический анализ и выведено уравнение влияния радиоактивного облучения на аутосомный геном с использованием междисциплинарных связей.
Ключевые слова: радиация, мутация, аутосомный геном, частота, разностное уравнение.

Hubal H.M. Mathematical analysis of the influence of radiation exposure on autosomal genome: interdisciplinary connections. The paper presents mathematical analysis and derives the equation of the influence of radiation exposure on the autosomal genome using interdisciplinary connections.
Keywords: radiation, mutation, autosomal genome, frequency, difference equation.

Вступ. Міждисциплінарні зв'язки необхідні як у наукових дослідженнях, так і в університетському навчанні майбутніх спеціалістів. Міждисциплінарні зв'язки найбільш ефективні, коли математики розуміють науки, знання яких необхідні для розв'язання поставленої задачі, а фахівці з напряму підготовки, до якого відноситься задана задача, вміють застосовувати набуті знання з вищої математики та інших необхідних для розв'язання поставленої задачі наук.

У наведеному нижче науковому дослідженні крім знань з вищої математики необхідні знання з екології, біології, фізики.

Основна частина. Зробимо математичний аналіз та виведемо рівняння впливу радіації на аутосомний геном у популяції.

Інтенсивний видобуток, одержання і використання радіоактивних елементів, недостатня система захисту, а також аварійні ситуації і ядерні випробовування – все це призводить до того, що підвищений радіоактивний фон являється джерелом мутацій, що зачіпає генофонди популяцій, у тому числі людської популяції. Це, у свою чергу, загрожує незворотними наслідками не тільки для теперішніх людей, але й для майбутніх поколінь.

Тому розглянемо існування і розвиток популяції в умовах дії на неї мутагенного радіоактивного випромінювання.

Англійський математик Г. Харді і німецький лікар Н. Вайнберг незалежно один від одного встановили закон про частотні співвідношення генотипів у популяції. Виявилось, що частотні співвідношення пари алельних генів у популяції відповідають формулі

$$(p+q)^2 = p^2 + 2pq + q^2 .$$

Таким чином, при схрещуванні різностатевих особин, за одне покоління, для аутосомного геному встановлюється рівновага Харді–Вайнберга. Згідно з цим законом генотипи *AA*, *Aa*, *aa* мають наступні частотні співвідношення:

$$(AA)p^2 : (Aa)2pq : (aa)q^2, \qquad (1)$$

де p – частота домінантного алельного гена (алеля) A,
q – частота рецесивного алельного гена (алеля) a.

З закону Харді–Вайнберга виходить, що:

- число гомозиготних домінантних особин у популяції дорівнює p^2N ;
- число гомозиготних рецесивних особин у популяції дорівнює q^2N ;
- число гетерозиготних особин у популяції дорівнює $2pqN$, де N – загальне число особин у популяції.

Отже,

$$N = p^2N + q^2N + 2pqN = N(p^2 + q^2 + 2pq),$$

звідки

$$p^2 + q^2 + 2pq = 1 \quad \text{або} \quad (p+q)^2 = 1.$$

Таким чином, розподіл генотипів у популяції залежить від частоти домінантного p і рецесивного q алельних генів.

За законом Харді–Вайнберга, у популяції особин, що вільно схрещуються, вихідне співвідношення в потомстві гомозигот (домінантних і рецесивних) і гетерозигот залишається постійним. Наприклад, у популяції, в якій розподіляється одна пара алельних генів *A* і *a*, особини будуть мати один із наступних трьох генотипів: *AA*, *Aa* або *aa*. Інші поєднання неможливі.

Отже, будь-яка популяція, у якій розподіл алельних генів *A* і *a* відповідає співвідношенню (1), знаходиться в стані генетичної рівноваги.

Оскільки кожен ген однієї алельної пари може бути *A* або *a*, то їхні частоти $p+q=1$. Отже, знаючи частоту одного гена, можна легко визначити частоту іншого, тобто якщо частота гена *A* дорівнює *p*, то частота гена *a* буде дорівнювати $q=1-p$.

Виведемо залежність частоти гетерозиготного генотипу від частоти рецесивного алельного гена. Для цього запишемо $2pq$ як функцію від *q*: $y = 2pq = f(q)$, звідки $p = \dfrac{y}{2q}$.

Оскільки $p+q=1$, то $(p+q)^2 = 1$, тобто $p^2 + 2pq + q^2 = 1$, звідки $2pq = 1-(p^2+q^2)$. Тоді, підставивши $p = \dfrac{y}{2q}$, одержимо:

$$y = 1 - \left(\frac{y^2}{4q^2} + q^2\right), \quad y^2 + 4q^2y + 4q^4 = 4q^2$$

або

$$(y + 2q^2)^2 = (2q)^2,$$

звідки, враховуючи, що $q \ge 0, y \ge 0$ (оскільки $p \ge 0$), одержимо:

$$y + 2q^2 = 2q$$

або

$$y = -2q^2 + 2q.$$

Графіком одержаної квадратної функції є парабола з вертикальною віссю симетрії. Загальний вигляд квадратної функції (квадратного тричлена): $y = ax^2 + bx + c$. Побудуємо графік одержаної квадратної функції.

Перша похідна $y' = -4q + 2$. Визначимо вершину параболи: розв'язуємо рівняння $y' = 0$, тобто $-4q + 2 = 0$, звідки $q = 0{,}5$. Тоді $y = -2q^2 + 2q = 0{,}5$. Отже, вершина параболи $(0{,}5;\ 0{,}5)$.

Друга похідна $y'' = -4 < 0$. Отже, парабола опукла.

В одержаному квадратному тричлені $c = 0$. Отже, парабола проходить через початок координат. Оскільки рівняння осі симетрії параболи $q = 0{,}5$ і ліва вітка параболи перетинає вісь *Oq* в точці $(0;\ 0)$, то права вітка параболи перетинає вісь *Oq* в точці $(1;\ 0)$. Зауважимо, що точки перетину параболи з віссю *Oq* можна знайти і так: розв'язуємо рівняння $-2q^2 + 2q = 0$, звідки $q_1 = 0,\ q_2 = 1$. Тоді точки перетину параболи з віссю *Oq*: $(0;\ 0)$ і $(1;\ 0)$.

Будуємо графік функції $y = -2q^2 + 2q$ (рис. 1).

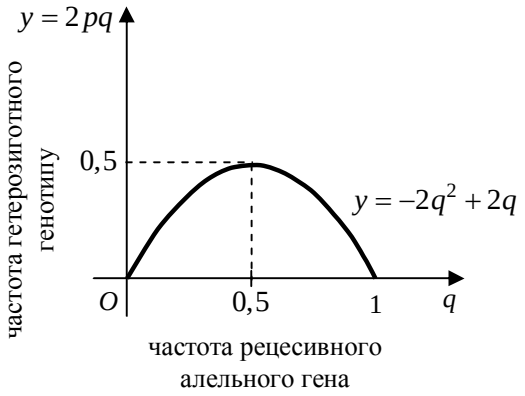


Рис. 1

Розподіл алелей у необмежено великій популяції при вільному схрещуванні, відсутності добору і без виникнення мутацій встановлюється на основі концентрації генів, які є в популяції. Частоту гена часто називають концентрацією гена. Частоту (концентрацію) гена часто виражають у процентах.

Співвідношення (1) залишається незмінним з покоління в покоління у випадку ідеальної популяції (число особин необмежено велике; немає добору, мутацій, міграції особин; існує панміксія і т.д.) [1]. Порушення співвідношення (1) за рахунок яких-небудь зовнішніх впливів: мутагенезу, міграції і т.д. – призводить тільки до зміни в наступному поколінні частотного співвідношення генів:

$$(A)p:(a)q,$$

відновлюючи співвідношення (1), тобто в наступному поколінні після схрещування буде новий зрівноважений розподіл генотипів. Постійна зміна станів рівноваги і перебудови генетичного складу популяції лежить в основі еволюції.

У процесі схрещування, яке відбувається під контролем природного добору, мутації набувають пристосовницького характеру – еволюція носить пристосовницький характер. Комбінації, що не забезпечують пристосування організму, усуваються добором. Навіть у бактерій і вірусів відбувається перекомбінація генетичного матеріалу, яка має таке саме біологічне значення, як і схрещування [2].

Одним із найбільш суттєвих факторів мутагенезу являється радіація. При цьому виникають порушення в генетичному апараті, які зазвичай несумісні з подальшим нормальним функціонуванням організму. Мутації генів в хромосомах можуть призвести або до зникнення репродуктивної функції в особини, або до появи у нащадків важких порушень.

Гени, зчеплені зі статтю, мають дещо більшу стійкість до радіоактивного впливу в порівнянні з аутосомними.

Зробимо математичний аналіз існування популяції і змін генофонду в умовах радіоактивної обстановки на прикладі зміни частот алелей, успадкованих аутосомно.

Нехай у процесі впливу на популяцію, частоти генотипів змінились в наступній пропорції:

$$(AA)(p^2+Fpq+\alpha):(Aa)[2pq(1-F)+\beta]:(aa)(q^2+Fpq+\gamma), \qquad (2)$$

що порушує співвідношення Харді–Вайнберга.

У співвідношенні (2) числа α, β, γ відображають вплив радіації відповідно на генотипи *AA*, *Aa*, *aa*; *F* – коефіцієнт інбридингу.

Зауважимо, що співвідношення (2) одержано з урахуванням, що

$$2pq(1-F)=2pq-2pqF=2pq-Fpq-Fpq.$$

У співвідношенні (2) умова панміксії ослаблена. Введена можливість близькородичевих шлюбів з коефіцієнтом інбридингу *F*. При інбридингу (у людини близькородичеві шлюби) ймовірність зустрічі гетерозигот, що виділяють шкідливі і летальні гени, різко зростає. Навіть при невеликій частоті небажаних гомозигот з шкідливими рецесивними генами число гетерозигот, які

є їх носіями, у популяції досить велике. Реально вплив на популяцію може бути пов'язаний з міграціями, виникненням субпопуляцій, радіоактивним опромінюванням і т.д.

Уже в наступному поколінні рівновага Харді–Вайнберга відновлюється з новим співвідношенням частот алелей. Для одержання частоти домінантного алеля *A* в новому співвідношенні необхідно додати частоту появи гомозиготи *AA* і половину частоти появи гетерозиготи *Aa* з співвідношення (2). Аналогічно і для алеля *a*.

Тоді одержимо:

$$(A)(p+\alpha+\beta/2):(a)(q+\gamma+\beta/2). \tag{3}$$

Таким чином, співвідношення частот генотипів в наступному поколінні:

$$(AA)(p+\alpha+\beta/2)^2:(Aa)2(p+\alpha+\beta/2)(q+\gamma+\beta/2): \\ :(aa)(q+\gamma+\beta/2)^2. \tag{4}$$

При цьому в (3) і (4) не входить коефіцієнт інбридингу *F*.

При відсутності впливів на популяцію $\alpha=\beta=\gamma=0$ співвідношення (4) тотожно (1), тобто маємо відоме положення про збереження частотного співвідношення генотипів у поколіннях.

При впливі на популяцію мутагенних факторів різної природи рівновага Харді–Вайнберга відновлюється, однак популяція не компенсує впливу – частоти алелів змінились. Тобто із типів рівноваги: стійка, нестійка, байдужа, рівновага Харді–Вайнберга – байдужа.

Нехай протягом довгого часу, поступово знижується радіоактивний вплив, в умовах якого розвивається популяція.

Розглянемо ситуацію, коли в регіоні існування популяції, на неї діє тільки один радіоактивний елемент з періодом піврозпаду *T*. Процент мутацій (зазвичай леталей) *r* (%) пропорційний потужності дози радіоактивного випромінювання. Враховуючи, що потужність дози радіоактивного випромінювання пропорційна активності радіоактивних елементів, розсіяних в навколишньому середовищі, можна записати, використовуючи основний закон радіоактивного розпаду,

$$r(\%):\ (1/2)^{(t/T)n},$$

де *t* років – приблизний час життя одного покоління, що існує в умовах радіації,
n – номер розглядуваного покоління, що існує в умовах радіації.

Приймаючи

$$(1/2)^{(t/T)}=R$$

і розглядаючи вплив на рецесивні гомозиготи *aa*, тобто $\frac{r(\%)}{100}=\gamma+\beta/2$, одержуємо:

$$\gamma+\beta/2=kR^n.$$

Ця рівність одержана введенням коефіцієнта *k*.

Коефіцієнт $k:\ N_0/T$, де N_0 – загальна початкова кількість радіоактивної речовини, що діє на популяцію. Коефіцієнт *k* характеризує початкову активність радіоактивних елементів, що впливають на популяцію.

З співвідношення (3) випливає рекурентна формула для зміни частоти алеля *a* в поколіннях для популяції:

$$q_n=q_{n-1}+\gamma+\beta/2=q_{n-1}+kR^n, \tag{5}$$

де $k<0$, $0<R<1$ (при $t=T$, $R=1/2$).

Розв'яжемо рівняння (5).

Перший спосіб. Запишемо рівняння (5) у вигляді:

$$q_i=q_{i-1}+kR^i,\quad i=1,2,\text{K},n.$$

Тоді

$$q_1 = q_0 + kR,$$

$$q_2 = q_1 + kR^2 = \left(q_0 + kR\right) + kR^2 = q_0 + kR(1+R),$$

$$q_3 = q_2 + kR^3 = [q_0 + kR(1+R)] + kR^3 = q_0 + kR(1+R+R^2),$$

$$q_4 = q_3 + kR^4 = [q_0 + kR(1+R+R^2)] + kR^4 = q_0 + kR(1+R+R^2+R^3),$$

К К К К К К К К К К К К К К

Отже,

$$q_n = q_0 + kR\sum_{i=1}^{n} R^{i-1}, \qquad (6)$$

де $\sum_{i=1}^{n} R^{i-1} = S_n$ – сума перших n членів геометричної прогресії. Тоді маємо $\sum_{i=1}^{n} R^{i-1} = \frac{1-R^n}{1-R}$. Таким чином, формула (6) набуде вигляду:

$$q_n = q_0 + k\frac{R(1-R^n)}{1-R}, \qquad (7)$$

де q_0 – частота алеля a в початковому поколінні, n – кількість поколінь.

Зауважимо, що при $n \to \infty$

$$\lim_{n\to\infty} S_n = \lim_{n\to\infty} \frac{1-R^n}{1-R} = \frac{1-\lim\limits_{n\to\infty} R^n}{1-R},$$

і враховуючи, що R додатне і менше 1, то $\lim\limits_{n\to\infty} R^n = 0$, і звідси випливає, що

$$\lim_{n\to\infty} S_n = \frac{1}{1-R}.$$

Другий спосіб. Запишемо $q_i = \lambda^i$, $i = 1,2,\text{K},n$, тоді $q_n = \lambda^n$, $q_{n-1} = \lambda^{n-1}$, де λ – невідома змінна.

Запишемо рівняння (5) у вигляді:

$$q_n - q_{n-1} = kR^n. \qquad (8)$$

Це рівняння – лінійне неоднорідне різницеве рівняння першого порядку зі сталими коефіцієнтами, в якому n – незалежна дискретна змінна [3]-[5]. Розв'язком q_n рівняння (8) є сума $q(n) + \text{??}(n)$ загального розв'язку $q(n)$ відповідного лінійного однорідного різницевого рівняння

$$q_n - q_{n-1} = 0 \qquad (9)$$

і частинного розв'язку $\text{??}(n)$ лінійного неоднорідного різницевого рівняння (8).

Загальний розв'язок лінійного однорідного різницевого рівняння шукаємо у вигляді:

$$q(n) = \lambda^n. \qquad (10)$$

Підставивши (10) в (9), одержимо:

$$\lambda^n - \lambda^{n-1} = 0.$$

Оскільки $\lambda^{n-1} \neq 0$, то поділивши обидві частини цього рівняння на λ^{n-1} одержимо характеристичне рівняння $\lambda - 1 = 0$, звідки $\lambda = 1$.

Тоді загальний розв'язок рівняння (9) має вигляд:

$$q(n) = C\lambda^n = C \cdot 1^n = C,$$

де C – стала величина.

Частинний розв'язок лінійного неоднорідного різницевого рівняння шукаємо за виглядом правої частини рівняння (8) у вигляді:

$$q(n) = \sigma R^n,$$

де σ – невідоме число.

Підставивши цю формулу в (8), одержимо:

$$\sigma R^n - \sigma R^{n-1} = kR^n.$$

Поділивши обидві частини цього рівняння на R^{n-1}, знайдемо σ:

$$\sigma = \frac{kR}{R-1}.$$

Тоді $q(n) = \frac{kR}{R-1} R^n.$

Таким чином, знаходимо розв'язок рівняння (8):

$$q_n = q(n) + q(n) = C + \frac{kR}{R-1} R^n. \qquad (11)$$

Сталу C знаходимо, виходячи з початкової умови: при $n = 0,\ q_n = q_0$.

Тоді, використовуючи формулу (11), одержимо:

$$q_0 = C + \frac{kR}{R-1} R^0 = C + \frac{kR}{R-1},$$

звідки $C = q_0 - \frac{kR}{R-1}.$

Тоді, підставивши C в (11), одержимо розв'язок рівняння (8) і відповідно рівняння (5):

$$q_n = q_0 - \frac{kR}{R-1} + \frac{kR}{R-1} R^n$$

або

$$q_n = q_0 + k\frac{R(1-R^n)}{1-R}.$$

Як бачимо, одержане другим способом рівняння співпадає з рівнянням (7), одержаним першим способом.

Приймаючи $q_n = 0$, можна, розв'язуючи рівняння (7) відносно n, одержати кількість поколінь, за які пройде повна елімінація алеля a, що знижує варіабельність генофонду популяції.

Значення періоду піврозпаду, вище якого рівновага Харді–Вайнберга не відновлюється внаслідок повної елімінації алеля a, називається критичним. Для великих періодів піврозпаду частота алеля a падає повільніше, але при цьому алель елімінує повністю за рахунок тривалого радіоактивного впливу.

Висновки. Таким чином, у статті зроблено математичний аналіз та виведено рівняння впливу радіоактивного опромінювання на аутосомний геном з використанням міждисциплінарних зв'язків.

1. Altukhov Yu.P. Genetic processes in populations / Yu.P. Altukhov. – M.: PTC "Academkniga", 2003. – 431 p.
2. Огурцов А.Н. Основы молекулярной биологии. – Ч. 1: Молекулярная биология клетки / А.Н. Огурцов. – Харьков: НТУ «ХПИ», 2011. – 304 с.
3. Годунов С.К. Разностные схемы / С.К. Годунов, В.С. Рябенький. – М.: Наука, 1977. – 440 с.
4. Шарковский А.Н. Разностные уравнения и их приложения: монография / А.Н. Шарковский, Ю.Л. Майстренко, Е.Ю. Романенко. – К.: Наукова думка, 1986. – 280 с.
5. Shun-Zhou Wu, Xiu-Min Zheng. Growth of solutions of some kinds of linear difference equations / Shun-Zhou Wu, Xiu-Min Zheng // Advances in difference equations. – 2015. – Vol. 142. – doi:10.1186/s13662-015-0485-8.

УДК 004:338.48
Сасюк З.К.
Національний університет водного господарства та природокористування

РОЗВИТОК ПРОСТОРОВОЇ УЯВИ СТУДЕНТІВ ТЕХНІЧНИХ ВНЗ МЕТОДАМИ НАРИСНОЇ ГЕОМЕТРІЇ

Сасюк З.К. Розвиток просторової уяви студентів технічних ВНЗ методами нарисної геометрії. У статті розглянуто проблему розвитку просторової уяви студентів вищих навчальних закладів методами нарисної геометрії. Запропоновано за допомогою послідовного показу графічних ілюстрацій підвищити легкість сприйняття багатоетапних геометричних побудов, активізувати просторове мислення студентів, спонукати їх до творчого пошуку різних рішень і подальшого вдосконалення графічних знань.
Ключові слова: просторова уява, наочність, геометричні побудови, просторове мислення, графічні знання.

Сасюк З.К. Развитие пространственного воображения студентов технических вузов методами начертательной геометрии. В статье рассмотрена проблема развития пространственного воображения студентов высших учебных заведений методами начертательной геометрии. Предложено с помощью последовательного показа графических иллюстраций повысить легкость восприятия многоэтапных геометрических построений, активизировать пространственное мышление студентов, побудить их к творческому поиску различных решений и дальнейшего совершенствования графических знаний.
Ключевые слова: пространственное воображение, наглядность, геометрические построения, пространственное мышление, графические знания.

Sasyuk Z.K. Development of spatial imagination of students of technical Universities methods of descriptive geometry. In this article the problem of spatial imagination university students methods of descriptive geometry. Proposed by sequential display graphic illustrations to increase ease of perception multistage geometric constructions, activate spatial thinking students, encourage them to find creative solutions and various further improvements graphic knowledge.
Keywords: spatial imagination, visualization, geometric construction, spatial thinking, graphic knowledge.

Оточуючий нас світ – це світ геометрії чистої, справжньої, бездоганної в наших очах. Все навколо – геометрія. Ніколи ми не бачили так чітко таких форм, як коло, прямокутник, кут, циліндр, куля, виконаних так виразно, з такою ретельністю і так впевнено. Ле Корбузьє

Постановка проблеми. Нарисна геометрія - інженерна дисципліна, з якої починається технічна освіта майбутнього інженера. Найважливішими завданнями нарисної геометрії є:

• навчити просторово мислити і відображати на площині тривимірні геометричні образи (фігури);

• розвинути здатність уявного сприйняття просторового геометричного образу за його відображенням на площині, тобто навчити читати креслення.

Відомо, що при вивченні нарисної геометрії у студентів виникають труднощі, пов'язані з особливим з'єднанням логічного мислення і просторової уяви, яке, за словами видатного російського геометра Н.А. Риніна, «є...таємничою і мало піддається вивченню точними науками здатністю людського духу...». Поєднання цих двох можливостей людського розуму створює новий рівень мислення - просторове мислення, яке дає можливість оперувати образами в просторі і без якого неможливі будь-яка інженерна діяльність, інженерна творчість і технічний прогрес.

У просторовому мисленні відбувається постійне перекодування образів, тобто перехід від просторових образів реальних об'єктів до їх умовно-графічних зображень, від тривимірних зображень до двомірних і назад. Графічний синтез зображень предмета на кресленику на основі графічної бази даних дозволяє зчитувати за допомогою графічного аналізу задану інформацію і включає роботу просторової уяви, об'єднуючи плоскі проекції предмета в його об'ємний цілісний образ. Ця складна розумова робота і є просторове мислення, розвиток якого і відбувається в процесі вивчення нарисної геометрії.

Аналіз останніх досліджень і публікацій. Науковцями доведено, що сформована база графічних опор і розвинене просторове мислення дозволяють скоротити процес графічного аналізу та синтезу зображень і створюють можливість швидкого і грамотного виконання та читання креслеників. Проблемі розвитку просторової уяви і просторового мислення в процесі графічної підготовки присвячено наукові праці та наукові дослідження В.Гордона, В.Михайленка, В.Сидоренка, М.Козяра, А.Корнєєвої. Психолого-педагогічні аспекти графічної підготовки

студентів досліджували такі вчені, як В.Ананьєв, А.Галкін, Є.Кабанов-Меллер, Н.Линьков, Б.Ломов, Л.Румянцев, В.Скакун, Ю.Трофимов, А.Умронходжаєв, И.Якиманська та інші. Питання практичного використання графічних знань при вивченні предметів загальноосвітнього та загальнотехнічного циклів розглядалися Н.Виноградовим, Л.Государським, Л.Левенбергом, С.Ковалевим, М.Макаровим, В.Михайленком, Л.Резниковим та ін.

Невирішені частини проблеми. Багато студентів не володіють досить розвиненим просторовим мисленням. Ця проблема є давньою, але актуальною. Засвоєння нарисної геометрії поряд з невмінням більшості студентів виконувати графічні логічні дії в розумовому просторі утруднюється також просторістю і новизною теоретичного і графічного ілюстративного матеріалу. За підручниками засвоїти предмет також непросто, тому що матеріал перевантажений пояснюючими графічними ілюстраціями та описами. У кожному навчальному посібнику по-своєму представлено наочність. У багатьох використовують зображення, виконані одним кольором, що дає недостатнє уявлення про простір і геометричні об'єкти.

Щоб активізувати і розвинути логічно-графічні властивості розуму і його можливості просторової уяви в процесі навчання необхідними є деякі зміни традиційної методики викладу курсу. Вирішенню завдань нарисної геометрії сприяє тематична модульна структуризація матеріалу нарисної геометрії з чіткими графічними характеристиками геометричних елементів, алгоритмізація та ілюстрація графічних дій за завданнями кожної теми.

У статті здійснено спробу розкрити роль графічних наочностей (ілюстрацій) при комплексному підході до вивчення деяких основоположних питань теорії нарисної геометрії для студентів технічних вузів.

Мета дослідження - за допомогою послідовного показу графічних ілюстрацій підвищити легкість сприйняття багатоетапних геометричних побудов, активізувати просторове мислення студентів, спонукати їх до творчого пошуку різних рішень і подальшого вдосконалення графічних знань.

Основні результати дослідження. Відомо, що наочність підвищує інтерес студентів до знань і робить процес навчання легшим. Багато теоретичних положень за вмілого використання наочності стають доступними і зрозумілими для студентів. К. Ушинський [3] відзначав: «Учіть дитину яким-небудь п'яти невідомим йому словам, і він буде довго і марно мучитися над ними; але зв'яжіть з картинками двадцять таких слів і дитина засвоїть їх нальоту...». Я. Коменский [4] наполягав на необхідності доводити студентів до розуміння зв'язку між явищами, процесами і так організовувати навчальний матеріал, щоб він не здавався студентам хаосом, а був би коротко викладеним у вигляді небагатьох основних положень та ілюстрацій.

Процес навчання спрощується при розумному використанні принципу наочності. Навчання не повинно бути перенасичене ілюстраціями та іншими формами наочності, але в деяких важкодоступних питаннях застосування наочності необхідне. Прикладом може слугувати завдання на перетин геометричних об'єктів, яке вирішують за допомогою виконання певних чітких алгоритмів, що значно полегшує виконання побудов лінії перетину геометричних об'єктів. Однак навіть маючи на кресленні вже вирішені завдання, студенту складно подумки уявити отриману лінію перетину заданих геометричних об'єктів. Усі психічні процеси, в тому числі і просторова уява, удосконалюються в результаті діяльності. Ця діяльність повинна чимось стимулюватися і направлятися, тобто необхідна система вправ [5, 6].

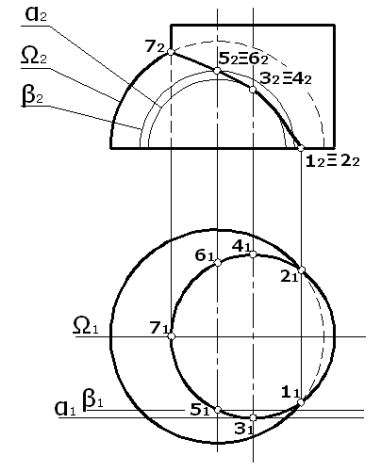
Традиційно в літературі [1, 2] лінію взаємного перетину поверхонь розглядають як множину точок спільних для обох поверхонь. ЇЇ визначають за точками перетину ліній однієї поверхні з іншою поверхнею або, що те ж саме, - за точками перетину ліній кожної із поверхонь.

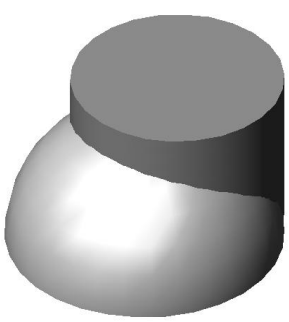
Для визначення точок, спільних для обох поверхонь, часто використовують допоміжні січні поверхні. Поверхні-посередники перетинають дані поверхні по лініях, які, у свою чергу, перетинаються в точках лінії перетину даних поверхонь.

Січні поверхні-посередники обирають так, щоб вони, перетинаючись із даними поверхнями, давали прості для побудов лінії, наприклад, прямі або кола. Часом застосовують і комбінацію допоміжних січних поверхонь.

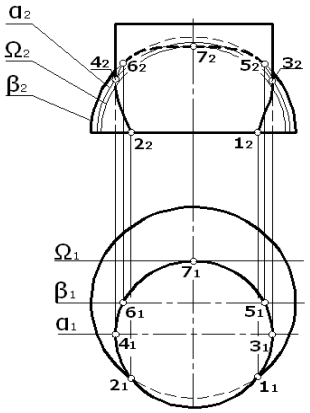
Із загальної схеми побудови лінії перетину поверхонь виділяють два основних способи – спосіб січних площин і спосіб січних сфер.

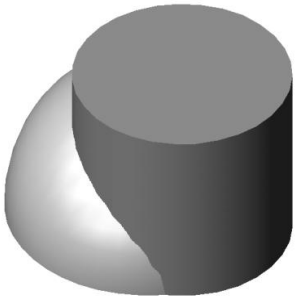
Сфера і циліндр є криволінійними поверхнями. Отже, лінією перетину буде просторова крива лінія, для побудови якої оберемо спосіб допоміжних січних площин-посередників. При ознайомленні зі способами побудови проекцій точок ліній взаємного перетину поверхонь шляхом введення площин-посередників необхідно розкрити особливості цих побудов, порівняти зображення, виділяючи спільне в них і відмінність.



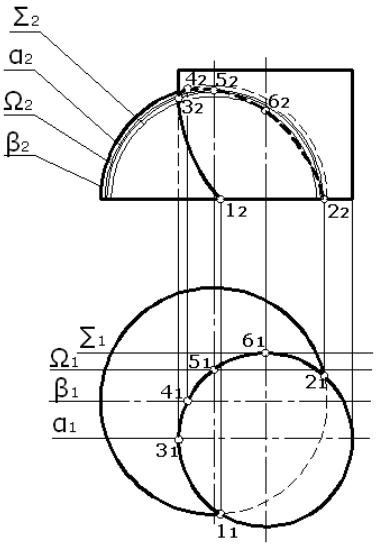


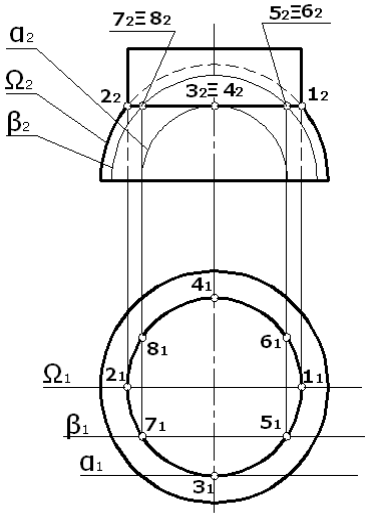
а)

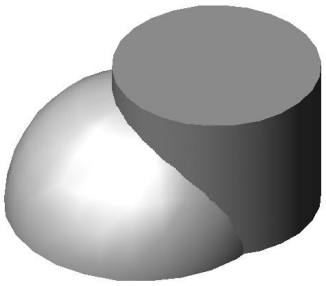




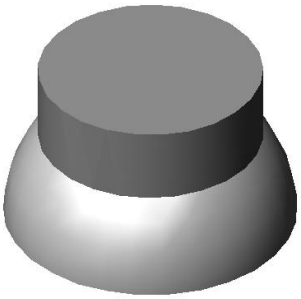
б)







в)



г)

Рис.1. **Взаємний перетин поверхонь сфери та циліндра**
[авторська розробка]

У цьому випадку важливо показати кілька ілюстрацій *(рис.1а-рис.1г)* з різним взаємним розміщенням циліндра відносно сфери: 1) осі сфери і циліндра належать спільній фронтальній площині *(рис.1а)*; 2) осі сфери і циліндра належать спільній профільній площині *(рис.1б)*; 3) осі сфери і циліндра належать різним площинам *(рис.1в)*; 4) осі сфери і циліндра збігаються, тобто поверхні співвісні *(рис.1г)*. Це відразу формує поняття про статичність і динамічність оригіналу простору, площин проекцій та геометричних об'єктів, показуючи, що об'єкти в просторі можуть змінювати своє положення, тим самим розширюючи просторові уявлення студентів.

Слід зауважити, що у всіх випадках поверхня циліндра займає горизонтально-проеціюче положення. Отже, горизонтальна проекція лінії перетину буде збігатися із горизонтальною проекцією циліндра і це буде коло.

Для побудови фронтальної проекції лінії перетину обираємо спосіб допоміжних фронтальних площин-посередників.

Важливо донести до відома студентів, що завжди побудову проекцій лінії перетину необхідно починати з характерних (опорних) точок, якими є найнижчі та найвищі точки, точки переходу видимого контуру лінії перетину в невидимий, найближчі та найдальші точки від спостерігача.

Так, на *рис. 1а - рис.1в* характерними є точки 1 та 2 перетину паралелі сфери з основою циліндра, оскільки вони знаходяться в одній площині і їх горизонтальні проекції перетинаються. Зауважимо, що на рис.1 фронтальні проекції точок 1 і 2 збігаються. Це зумовлено симетричним розміщенням зазначених точок відносно осі. Також необхідно проаналізувати і порівняти видимість точок 1 і 2 при зміні положення циліндра, що спонукає студентів до низки розумових і пізнавальних дій щодо взаємного розміщення точок як об'єктів простору.

Наступним кроком є визначення найвищих точок лінії перетину. При зміні положення циліндра положення найвищої точки також буде змінюватися. На *рис.1а* це точка 7, яка знаходиться на перетині твірної обрису циліндра і меридіана сфери максимального радіусу, які належать фронтальній площині Ω. На *рис.1б* найвищою буде проекція точки 7, яку визначають за допомогою фронтальної площини Ω. На *рис.1в* найвищою є проекція точки 4, яку визначають за допомогою площини-посередника β.

Наступними опорними точками будуть точки переходу видимого контуру лінії перетину в невидимий. На *рис.1а* лінія перетину є симетричною відносно фронтальної площини Ω, тому фронтальні проекції видимого та невидимого контуру лінії перетину збігаються. В інших випадках *(рис.1б-рис.1в)* межею видимості слугують обрисові твірні циліндра, оскільки поверхня циліндра знаходиться ближче до спостерігача. Відзначаємо точки 3 та 4 *(рис.1б)* та точку 3 *(рис.1в)* переходу видимого контуру в невидимий контур лінії перетину, які визначають за допомогою площини-посередника α.

Точки 5 і 6 є допоміжними точками, які необхідні для більшої точності побудови лінії перетину. Їх положення визначаємо за допомогою площин-посередників β *(рис.1а-рис.1б)*, Ω та ∑ *(рис.1в)*.

Особливої уваги заслуговує випадок, коли осі сфери і циліндра збігаються, тобто поверхні співвісні *(рис.1г)*.Відомою є така властивість поверхонь обертання: дві будь-які співвісні поверхні

обертання перетинаються по колах, які проходять через точки перетину меридіанів поверхонь. Ці кола лежать в площинах перпендикулярних до осі поверхонь обертання. Якщо одна із поверхонь є сфера, то для неї будь-який діаметр можна прийняти за вісь обертання. Отже, сфера з центром на осі обертання перетинає поверхню циліндра по колу. Вісь поверхонь обертання паралельна фронтальній площині проекцій, то на цю площину лінія перетину проеціюється у відрізок прямої лінії.

Аналіз і відображення опорних точок лінії перетину на площину проекцій при різних положеннях циліндра супроводжуються рухливістю просторово-образного мислення, розвитком пізнавального мислення студентів. Щоб студенти могли об'єднувати плоскі проекції предмета в його об'ємний цілісний образ, необхідно супроводжувати показ об'ємними моделями. Це однозначно допомагає здійснювати графічний синтез зображень предмета і включає роботу просторової уяви. Студент чітко бачить перехід від просторових реальних образів до їх умовно-графічних зображень, від тривимірних до двомірних і назад.

З одного боку, це забезпечує студентам можливість сприйняття схожих зразків з усіма їх конструктивними особливостями, а з іншого боку - дозволяє при предметному (образному) спогляданні, обговорити і усвідомити ці конструктивні особливості та виявити технічні проблеми, отримати потрібні навики графічних побудов.

Такий метод подання матеріалу дозволяє підвищити навчальну мотивацію студентів, сприяє розвитку їх пізнавальної активності, підвищує легкість сприйняття багатоетапних геометричних побудов. Разом з тим, просторове мислення дозволяє студентам будь-якого рівня активно включитися в навчально-пізнавальний процес і максимально проявити себе: заняття можуть проводитися на високому рівні складності, але включати в себе питання, доступні та цікаві всім.

Висновки. Нарисна геометрія є однією з фундаментальних навчальних дисциплін, що розвивають наочно-образне мислення. У цих умовах велике значення має визначення того, які з нових методів навчання дають найбільший ефект при викладанні нарисної геометрії та подальше впровадження їх у навчальний процес.

Рівень графічної підготовки студента зараз визначається не стільки технікою графічних зображень, а тим, наскільки він готовий до розумового перетворення цих зображень і наскільки розвинена рухливість образного мислення, а також рівень просторових уявлень, які є одним з показників загального розумового розвитку. Графічна діяльність вимагає виконання низки розумових і пізнавальних дій, якісне втілення яких здійснюється за наявності в студентів здібностей до сприйняття різних засобів графічної інформації, її переробки, переосмислення, аналізу цілісності сприйняття. Однак, якого б ступеню розвитку логічного мислення не досягли студенти, наочність завжди буде найважливішим засобом їх навчання комплексу дисциплін «Нарисна геометрія», «Інженерна графіка». Принцип наочності проявляється не тільки у використанні наочності, як засобу ілюстрації, але і в орієнтації студента на самостійну роботу з образом, особливо графічним, як джерелом інформації, що містить в собі і загальне, і одиничне, і особливе, і індивідуальне.

Використання даної методики сприяє розвитку у студентів необхідних просторових уявлень, де одночасно формуються і взаємодіють як статичні, так і динамічні компоненти. Все це підвищує культуру геометро-графічної підготовки студентів, тим самим створюючи необхідну базу для подальшого вивчення курсу нарисної геометрії.

1. 1.ГордонВ.О.Курс нарисної геометрії: навч. посібник для втузів/В.О.Гордон,М.А.Семенцов-Огієвський. - М.: Вища школа, 2000. -272 с.
2. 2.КорольовЮ.І. Нарисна геометрія: підручник для вузів /Ю.І.Корольов. - СПб: Лань, 2008. - 252с.
3. Ушинский К.Д.Педагогическиесоч. в 6-ти т. — Т. 2. — М., 1988. — С. 30
4. Коменский Я.А. «Великая дидактика» (избр. главы (по хрестоматии). - М.: Просвещение, 1988.
5. Зелёный П.В., Белякова Е.И., Лифанова О.А. Роль начертательной геометрии в общепрофессиональной подготовке инженера. Материалы международной научно-практической конференции. Инновационные технологии в инженерной графике. Проблемы и перспективы. Брест, 2014.- С.47-49.
6. Житенева Н.С., Яромич Н.Н. Анализ эффективности методов 3d-моделирования Материалы международной научно-практической конференции. Инновационные технологии в инженерной графике. Проблемы и перспективы. Брест, 2014.- С.72-74.