Biomedical Engineering and Technology
Issue 17(1), 2025 ISSN (Online) 2707-8434

УДК 07.11.07-09

ПОРІВНЯЛЬНИЙ АНАЛІЗ ГРАФІЧНИХ ТЕХНОЛОГІЙ WEB CANVAS ТА POSTSCRIPT (ОГЛЯД)

Соломін Андрій Вячеславович[1]
a.solomin@kpi.ua
Гетун Галина В'ячеславівна[2]
galinagetun@ukr.net
Богачук Владислав Віталійович[1]
bohachuk.vladyslav@lll.kpi.ua
Борисенко Роман Володимирович[1]
borysenko.roman@lll.kpi.ua
[1]Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», м. Київ, Україна
[2]Київський національний університет будівництва і архітектури, м. Київ, Україна

Анотація – *Наразі стрімко розвиваються цифрові графічні технології, поступово заміщуючи класичні поліграфічні паперові аналоги. Але поступово помираюча зараз поліграфія пройшла довгий шлях у своєму розвитку і накопичила багатий досвід, у тому числі у сенсі комп'ютерної обробки інформації. Цей досвід дуже корисно використовувати зараз для удосконалення сучасних цифрових технологій, а аналіз результатів такого використання створює базу для їх подальших збагачень. У роботі було проведено порівняльний аналіз сучасної графічної web-технології canvas та найбільш досконалої цифрової поліграфічної технології PostScript. На основі виявлених аналогій проілюстровано тенденцію розвитку сучасних технологій та запозичення досвіду їх попередників. Схожість підходів класичних і сучасних засобів обробки і представлення графічної інформації проілюстровано на прикладах використання методів тимчасового збереження та відновлення станів графічної системи, поточної трансформаційної матриці з однорідними координатами, що дозволяє суттєво оптимізувати трансформації графічних об'єктів, представлення графічної інформації у вигляді комп'ютерних програм на відміну від статичних файлів. Деякі із запозичених методів використовуються наразі і в галузі тривимірної графіки, а поширення їх на інші галузі обробки графічної інформації досить перспективно.*

Ключові слова*: комп'ютерна графіка, системи відображення, веб-технології, програмна інженерія, технологія створення програмних продуктів*

I. ВСТУП

В XXI столітті спостерігається стрімкий перехід від паперових до цифрових засобів зберігання та передавання інформації. Класичні друкарні книжок, газет, журналів фактично зникають, натомість з'являються цифрові платформи, які переймають функції своїх попередників. Головним інструментарієм цифрових інформаційних технологій, що заміщують поліграфічні, наразі є Web-технології. Тому дуже перспективно проаналізувати досягнення класичних поліграфічних засобів у порівнянні з їх Web-нащадками, окреслюючи тенденції їх розвитку.

II. МЕТА ДОСЛІДЖЕННЯ

Мета роботи – виявити загальні тенденції розвитку графічних технологій на основі порівнянь класичних поліграфічних засобів відображення та сучасних Web-технологій.

III. WEB-ТЕХНОЛОГІЯ CANVAS – НАЩАДОК ЗАСОБІВ POSTSCRIPT

Web-технології з'явились та інтенсивно розвиваються в останні десятиліття, і саме в цей період поступово нівелюється класична паперова поліграфія, передаючи свої надбання цифровому нащадку. Історія розвитку поліграфії налічує тисячоліття, і накопичений досвід плідно використовується зараз у Web. Корисно відслідкувати деякі аналогії, бо як завжди, на стику технологій виникають нові ідеї.

3.1. PostScript в класичній поліграфії

Отже, історія технологій класичної поліграфії дуже давня і насичена різними

досягненнями, але навіть після появи комп'ютерних засобів обробки інформації відбулась зміна кількох поколінь відповідних програмних продуктів для верстки поліграфічних видань – від Ventura Publisher через Aldus PageMaker, QuarkXPress до Adobe InDesign. Останній вважається вершиною, досягнутою в галузі поліграфічних комп'ютерних технологій і базується на розробленій в Adobe Systems спеціальній мові програмування PostScript [1-4]. Головними проблемами всіх попередніх варіантів були нестабільність результатів верстки при перенесенні файлів з комп'ютера видавництва до комп'ютера друкарні і не повна відповідність прев'ю малюнків на екрані з їх відтворенням на друкарських верстатах. Технологія PostScript вирішила ці проблеми завдяки заміні статичних графічних файлів різних форматів комп'ютерними програмами з використанням спеціально розробленої мови PostScript. Таким чином, тепер файли з текстами і зображеннями не відтворюються безпосередньо на екранах і друкарських засобах, а виконуються (інтерпретуються) як комп'ютерні програми інтерпретаторами PostScript з використанням відповідних драйверів, які враховують особливості систем відображення.

Багато ідей, які лежать в основі PostScript, тепер застосовується в сучасних Web-технологіях.

Власне PostScript (PS) – типова інтерпретована програмна мова, що базується на стеках та оперує з чотирма основними стеками.

Стек операндів – це спеціальний стек, в який поміщаються аргументи (операнди) операторів PS перед їх виконанням, а результати виконання операторів повертаються на вершину цього стека.

Наприклад, для перемножити двох аргументів 124 і 236 використовується наступний PS-код:

```
124 236 mul
```

Перші два числа, «124» і «236», додаються в стек операндів. Оператор «mul» виконує множення, вилучаючи два останніх числа зі стека операндів, перемножує їх і повертає результат у той самий стек на його вершину. Результат може залишатися в стеку для подальшого використання іншими операторами в програмі.

Стек словників – це набір всіх відкритих на даний момент словників. Словник складається з пар "ім'я (key)-значення", де зберігаються всі змінні та їх значення. Також всі оператори мови зберігаються в словниках разом з їх кодами для виконання. У PostScript немає зарезервованих слів, включаючи імена вбудованих операторів, тобто програма може змінювати імена операторів або додавати нові.

Коли в програмі зустрічається будь-яке ім'я (key), інтерпретатор шукає його першу появу серед усіх словників у стеку, починаючи з його вершини. Імена асоціюються зі змінними, операторами та процедурами.

У стеку словників завжди присутні такі словники:

Systemdict: системний словник з атрибутом "тільки читання", де зберігаються всі імена PS-операторів та їх коди.

Userdict: користувацький словник з атрибутом "доступний для запису", де зберігаються змінні, що використовуються PS-програмами.

Словник Userdict знаходиться на вершині стеку постійно присутніх словників. Оператор def, який створює нові процедури, додає визначення саме сюди. Розташування Userdict на вершині стеку дозволяє перевизначати будь-які оператори PostScript, навіть якщо відповідний оператор в Systemdict має атрибут "тільки читання". Це можливо завдяки тому, що інтерпретатор шукає імена, починаючи з вершини стеку, і якщо ім'я знайдено в Userdict, пошук далі не продовжується.

Стек виконання (execution) використовується для зберігання об'єктів, що знаходяться в процесі виконання, таких як процедури. Спочатку процедури, які необхідно виконати, поміщаються в цей стек, виконуються поетапно і потім видаляються зі стеку. Якщо інтерпретатору необхідно призупинити виконання будь-якої процедури, щоб розпочати нову, він поміщає нову процедуру на вершину стеку, виконує її, а потім видаляє зі стеку. Після

цього відкладена процедура знову опиняється на вершині стеку та продовжує виконання.

Стек станів графічної системи (graphics state) зберігає копії значень спеціальної структури даних під назвою «поточний стан графічної системи» (graphics state). У цій структурі міститься інформація про поточний колір ліній та заповнення графічних об'єктів, товщину ліній малювання, шрифт, трансформаційну матрицю та інші параметри. Більшість з цих параметрів керується окремо за допомогою операторів мови PostScript. Стек використовується для тимчасового зберігання всіх цих параметрів за допомогою оператора gsave та подальшого відновлення за допомогою оператора grestore.

Доцільно підкреслити тут корисність ідеї використання згаданої вище трансформаційної матриці, як складової поточного стану графічної системи. Ця технологія зараз використовується всюди, де виникає потреба всіляких трансформацій графічних об’єктів. Її сутність полягає у застосуванні матриць для здійснення поворотів, масштабувань, віддзеркалень та переміщень об’єктів. Але якщо повороти, масштабування та віддзеркалення реалізуються звичайними матрицями, наприклад, для операцій на площині матрицями 2x2, то для переміщень використовуються так звані однорідні координати. У цьому випадку штучно додається ще одна координата, трансформаційні матриці стають розміром 3x3, але це дозволяє відобразити переміщення об’єктів також через матриці.

Функціонування PS-інтерпретатора ґрунтується на наступному принципі. На вхід інтерпретатора надходить потік символів (в ASCII-кодуванні) або потік бітів (у binary-кодуванні), які являють собою PS-опис сторінок. Інтерпретатор сканує цей потік, розбиває його на об'єкти згідно з синтаксисом мови PostScript, визначає їх тип і відповідно до цього типу виконує дії. Літеральні об'єкти, такі як числа або строкові дані, просто додаються на вершину стека операндів. Іменовані об'єкти інтерпретатор шукає у словниках (dictionary), які відкриті та згадані у поточний момент у стеку словників. Знайдене у словнику значення виконується, якщо це оператор або процедура, або додається до стеку операндів, якщо це літеральний об’єкт.

В технології PostScript окремо розглядаються простори користувача і вивідного пристрою.

Простір вивідного пристрою (device space) – це система координат, яку розуміє вивідний пристрій, і одиниці вимірювання якої зазвичай відповідають його роздільній здатності. Цей простір використовується на останньому етапі – виведення графічної інформації.

Простір користувача (user space) – це координатна система, яка використовується PS-програмами для опису розташування графічних об’єктів. По суті, це аналог першого квадранта звичайної прямокутної системи координат з початком відліку у лівому нижньому куті. Координати виражаються дійсними числами, тому поняття роздільної здатності в цьому просторі не має сенсу. Інтерпретатор конвертує простір користувача в простір вивідного пристрою за допомогою відповідного PS-драйвера з використанням поточного стану графічної системи, у тому числі поточної трансформаційної матриці.

Поточна трансформаційна матриця є одним з елементів структури даних «поточний стан графічної системи» (graphics state). Іншими її елементами є, наприклад, товщина, колір і тип ліній, колір заповнення замкнених областей на зображеннях, різновид та розмір шрифту і т.ін.

Для малювання ліній спочатку створюється шлях (path). Це сукупність прямих ліній і кривих, розташованих на сторінці. Сам по собі шлях є уявним і не наноситься безпосередньо на папір принтера. Оператор stroke використовується для перетворення цього уявного шляху на видимий малюнок на папері, визначаючи товщину, колір і тип лінії відповідно до параметрів у поточному стані графічної системи на момент виконання.

Якщо шлях є замкнутим, область, обмежену ним, можна заповнити кольором за допомогою оператора fill, або вирізати,

зробивши невидимими частини зображень поза межами шляху, за допомогою оператора clip. При заповненні області кольором за допомогою оператора fill, колір і щільність фарби визначаються параметрами, зазначеними в поточному стані графічної системи.

3.2. Canvas у Web-технологіях

Тепер щодо графічних Web-технологій. Тут також пройдено значний шлях удосконалень і вирішень проблем. За деякою аналогією з поліграфією складнощі пов'язані в першу чергу з сумісністю пристроїв та браузерів, як засобів відтворення. Адже Web-сторінки можуть переглядатись і на мобільних телефонах, і на широких моніторах. Наприклад, для широких моніторів з'явилась потреба верстки текстів у вигляді колонок, як для газет класичної поліграфії, щоб не «бігати» очима по довгих рядках тексту. Отже, і тут спостерігається залучення ідей класичної поліграфії для розвитку графічних Web-технологій.

«Вершиною» в історії розвитку технологій Web-дизайну наразі є html 5. В реалізації цієї версії дуже багато можливостей, які раніше не були стандартизованими і створювались по-різному в різних браузерах, іноді неоднозначно і нестабільно. Однією з них є можливість створення графічних елементів на Web-сторінках безпосередньо без залучення спеціальних графічних редакторів за допомогою елемента canvas [5-8]. Детальний аналіз його функціональності свідчить про залучення ідей PostScript, які в свою чергу є «вершиною» в історії розвитку класичних поліграфічних технологій.

Прослідкуємо аналогії у використанні canvas і PostScript на прикладі створення логотипу засобами об'єктів елементу canvas.

Отже приклад створеного на Web-сторінці логотипу виглядає наступним чином (Рис. 1).

Рисунок 1 – Приклад логотипу

Пропустивши загальні відомі всім html-елементи web-сторінки, наведемо тут фрагменти коду, що ілюструють зазначені вище аналогії (номери рядків наведено для зручності посилань на них):

```
1. <canvas  id="logo" width="1000"
   height="100">
2. </canvas>
3. <script>
4. var canvas =
   document.getElementById('logo');
5. if (canvas.getContext){
6. var context = canvas.getContext('2d');
7. }else{alert("canvas error");
8. };
9. context.strokeStyle ="#0000FF";
10. context.fillStyle ="#0000FF";
11. context.lineWidth = 2;
12. context.beginPath();
13. context.moveTo(0, 40);
14. context.lineTo(30, 0);
15. context.lineTo(60, 40 );
16. context.lineTo(300, 40 );
17. context.stroke();
18. context.font = "italic 40px 'Arial'";
19. context.fillText ("Biomedical", 75, 35);
20. context.save();
21. context.strokeStyle ="#00FFFF";
22. context.fillStyle ="#00FFFF";
23. context.translate(20,20);
24. context.fillRect(0,0,20,20);
25. context.fillStyle = "#FFFF00";
26. context.strokeStyle = "#FFFF00";
27. context.lineWidth = 2;
28. context.beginPath();
29. context.moveTo(0, 20);
30. context.lineTo(10, 0);
31. context.lineTo(20, 20);
32. context.lineTo(0, 20);
33. context.closePath();
34. context.fill();
35. context.restore();
36. </script>
```

Тут в рядках 1-2 створюється елемент canvas із зазначенням id-селектора та розмірів. Далі рядки 3-36 відображують контейнер зі сценарієм JavaScript, що реалізує створення вказаного логотипу. В рядках 5-8 після перевірки створюється екземпляр об'єкту «контекст малювання» та привласнюється змінній context. Далі всі дії

відбуваються з властивостями і методами цього об'єкту.

В рядках 9-11 задаються параметри «поточного стану графічної системи», а саме колір і товщина ліній та колір заповнення замкнених графічних елементів (синій). В рядках 13-16 створюється віртуальний шлях трикутного контуру з горизонтальною лінією, а в рядку 17 цей шлях методом stroke() перетворюється в реальну графічну лінію з параметрами, зазначеними в «поточному стані графічної системи». В рядку 18 задаються параметри шрифту, а в рядку 19 створюється фрагмент тексту з цими параметрами. Ці дії є повними аналогами відповідних технологій PostScript.

Далі в рядку 20 методом save() «поточний стан графічної системи» зберігається в стеку, щоб потім відновитися в рядку 35 методом restore().

В рядках 21-23 зазначаються нові параметри «поточного стану графічної системи», а саме блакитний колір створюваного далі в рядку 24 прямокутника, та перенесення початку координат методом translate(20,20) в рядку 23. Прямокутних малюється відносно нового початку координат. Таке перенесення початку координат пояснює призначення і переваги збереження і відновлення «поточного стану графічної системи». Адже це дозволяє створювати незалежно окремі графічні елементи і розміщувати їх у відповідних місцях сторінки. Після цього початковий стан сторінки відновлюється, і на ній можна створювати нові графічні елементи.

Аналогічно далі в рядках 25-27 задається новий «поточний стан графічної системи» з жовтим кольором ліній та заповнень, а в рядках 28-34 створюється жовтий трикутник, який накладається на раніше створений блакитний прямокутник.

Акцентуємо ще раз додатково, що в складі структури «поточного стану графічної системи» є «поточна трансформаційна матриця», в якій використовуються «однорідні координати» і яка дозволяє виконувати будь-які трансформації відповідного графічного елементу.

Більш детально (для двовимірного простору). Будь-які афінні перетворення у двовимірному просторі описуються співвідношеннями

$$\begin{aligned} x^* &= \alpha x + \beta y + \lambda, \\ y^* &= \gamma x + \delta y + \mu, \end{aligned} \tag{1}$$

де $\alpha, \beta, \gamma, \delta, \lambda, \mu$ – параметри цього афінного перетворення. Звичайними матрицями 2x2 можна відобразити будь-які з цих перетворень, крім перенесень (трансляцій), які описуються параметрами λ і μ.

Наприклад, обертання на кут φ описується матричним перетворенням

$$(x^*, y^*) = (x, y)\begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \tag{2}$$

Щоб забезпечити можливість матричного представлення всіх афінних трансформацій, у тому числі трансляцій графічних елементів, використовується технологія однорідних координат, яка полягає у штучному додаванні ще однієї координати і перетворенні двовимірних матриць трансформацій у тривимірні. Третя, штучна, координата дорівнює просто одиниці, а довільне матричне перетворення має вигляд:

$$(x^*, y^*, 1) = (x, y, 1)\begin{pmatrix} \alpha & \gamma & 0 \\ \beta & \delta & 0 \\ \lambda & \mu & 1 \end{pmatrix} \tag{3}$$

Якщо розкрити цей матричний вираз, то отримаємо рівняння афінних перетворень (1) і тотожність: 1=1.

Виявляється, що послідовність будь-яких афінних трансформацій в однорідних координатах можна представити, як результат перемноження відповідних матриць. Переваги цієї можливості у тому, що замість послідовного перетворення відповідними матрицями (масштабування, повороти, трансляції) кожної точки графічного об'єкту можна створити

результуючу матрицю всіх перетворень, а потім її застосовувати до всіх точок об'єкта. Звичайно, це суттєва економія ресурсів комп'ютера.

Слід зазначити, що ця технологія не обмежується лише двовимірним простором, а може бути використана і для трансформацій 3D-об'єктів. Звичайно, тоді штучний простір буде 4-вимірним, а трансформаційні матриці – 4x4.

Таким чином на сторінці (або в тривимірному просторі) можна розміщувати будь-яку кількість незалежних графічних об'єктів, довільно трансформуючи їх.

Це спільна риса класичної поліграфічної технології PostScript та сучасної web-технології canvas. Можна прогнозувати, що цифрові технології будуть і надалі використовувати надбання своїх попередників, які досягли своєї досконалості в результаті довгої еволюції.

Слід зазначити, що файли формату pdf – це спрощений варіант технології PostScript, тобто це результат виконання коду програми PostScript інтерпретатором віртуального принтеру, який зазвичай є складовою будь-якої операційної системи.

IV. ВИСНОВКИ

Розглянуто у порівнянні сучасні технології відображення графічної інформації в галузі класичної поліграфії та Web-дизайну. Виявлені аналогії свідчать про загальні тенденції запозичень нащадками позитивного досвіду, накопиченого в процесі розвитку своїми технологіями-попередниками. У наведеному порівняльному аналізі наведено лише найбільш яскраві та технологічно досконалі приклади плідних рішень на стику різних галузей, особливо технологій з довгою історією та сучасних, що поступово заміщують їх. Таких прикладів насправді багато, і не всі надбання застарілих технологій використано в плані удосконалення сучасних. Тому перспективно проводити їх порівняльні аналізи, що сприятиме подальшому прогресу.

ORCID ID та внесок авторів.

0000-0002-5226-8813 (A, D, F) Andriy Solomin

0000-0002-3317-3456 (D, E) Galina Getun

0009-0009-9263-6041 (B, D) Vladyslav Bohachuk

(C, D) Roman Borysenko

A - Концепція роботи та дизайн;

B - Аналіз даних;

C - Відповідальність за статистичний аналіз;

D - Написання статті;

E - Критичний огляд;

F - Остаточне схвалення статті.

ПЕРЕЛІК ПОСИЛАНЬ

1. Adobe Systems Incorporated. "PostScript Language Reference Manual". Addison-Wesley Publishing Company, 1999. URL: https://printtechnologies.org/standards/files/postsciptlrm.pdf?form=MG0AV3

2. Brundage, Steven M. "Learning PostScript by Doing: A Tutorial". No Starch Press, 1990.

3. Crawford, William C. "PostScript Explained". Addison-Wesley Publishing Company, 1987.

4. Davis, Jim. "Inside PostScript". M&T Books, 1988.

5. Smith, J. Interactive Web Design with Canvas. New York: Web Publishing Inc., 2023.

6. Brown, L. Advanced Canvas Techniques. Boston: TechPress, 2022.

7. Johnson, M. Canvas and JavaScript: A Practical Guide. San Francisco: CodeCrafters, 2021.

8. Williams, R. Building Responsive Web Apps with Canvas. Chicago: Digital Media Group, 2020.

Biomedical Engineering and Technology
Issue 17(1), 2025 ISSN (Online) 2707-8434

UDC 07.11.07-09

COMPARATIVE ANALYSIS OF GRAPHIC TECHNOLOGIES WEB CANVAS AND POSTSCRIPT (REVIEW)

Andriy Solomin[1]
a.solomin@kpi.ua
Galina Getun[2]
galinagetun@ukr.net
Vladyslav Bohachuk[1]
bohachuk.vladyslav@lll.kpi.ua
Roman Borysenko[1]
borysenko.roman@lll.kpi.ua
[1]National Technical University of Ukraine
«Igor Sikorsky Kyiv Polytechnic Institute»,
Kyiv, Ukraine
[2]Kyiv National University of Construction and Architecture,
Kyiv, Ukraine

Abstract - *Digital graphic technologies are currently developing rapidly, gradually replacing classic printing paper analogs. However, the now gradually declining printing industry has come a long way in its development and has accumulated a wealth of experience, including in the context of computer data processing. This experience is very useful now for improving modern digital technologies, and the analysis of results of such use creates a basis for their further enhancement. The work conducted a comparative analysis of modern web canvas technology and the most advanced digital printing technology PostScript. Based on the identified analogies, the trend of development of modern technologies and the borrowing of experience from their predecessors is illustrated. The similarity of approaches between classic and modern means of processing and presenting graphic information is illustrated by examples of using methods for temporarily saving and restoring the states of graphics system, the current transformation matrix with homogeneous coordinates, which allows for significant optimization of graphic objects' transformations, and presenting graphic information in the form of computer programs rather than static files. Some of the borrowed methods are now also used in the field of 3D graphics, and their extension to other areas of graphic information processing is quite promising.*

Keywords: *computer graphics, display systems, web technologies, software engineering, software product development technology*