кольорових зображень. Однак, незалежно від типу зображення пропорційно до розмірності вхідного зображення може зрости розмір шифрованого зображення.

Вказаний алгоритм можна використати для передачі графічних зображень.

1. Шнайер Б. Прикладная криптография. – М.: Триумф, 2003. – 815 с. 2. Яне Б. Цифровая обработка изображений. – М.: Техносфера , 2007. – 583 с. 3. Рашкевич Ю.М., Пелешко Д.Д., Ковальчук А.М., Пелешко М.З. Модифікація алгоритму RSA для деяких класів зображень. Технічні вісті 2008/1(27), 2(28). – С. 59–62. 4. Rashkevych Y., Kovalchuk A., Peleshko D., Kupchak M. Stream Modification of RSA Algorithm For Image Coding with precize contour extraction. Proceedings of the X-th International Conference CADSM 2009. 24-28 February 2009, Lviv-Polyana, Ukraine, Pp. 469–473.

УДК 004.83; 004.89

Я. Ковівчак, Ю. Кущев
Національний університет “Львівська політехніка”,
кафедра автоматизованих систем управління

ПРОГРАМНИЙ ЗАСІБ ВІДОБРАЖЕННЯ ПРОСТОРОВО-ЧАСОВИХ ДАНИХ

Наведено основні можливості та функціональні особливості програмного продукту опрацювання та візуалізації просторово-часових даних. Показано переваги розробленого програмного пакета порівняно з наявними засобами побудови відображень просторових даних.

The basic features and functional features of the software processing and visualization of spatial-temporal data is proposed. The advantage of developed software package to existing means of constructing maps of spatial data.

Вступ

Використання сучасної техніки і новітніх інформаційних технологій у повсякденній діяльності людини пов’язане з опрацюванням великої кількості даних. Цей процес переважно відбувається без безпосередньої участі людини, і його кінцевою метою є практична реалізація необхідного набору функцій відповідного технічного обладнання.

З розвитком науки і техніки зростає складність проблем, які необхідно розв’язувати, а отже, і обсяги оперування даними. Для полегшення цього процесу дані подаються в наочній формі за допомогою графіків, діаграм, багатовимірних залежностей. Це спрощує аналіз даних, отримання потрібної інформації та прийняття необхідних рішень.

Всі процеси у навколишньому світі мають часовий вимір, тому отримані і опрацьовані дані відповідають тільки певному поточному моменту часу і характеризують процес лише відповідно до нього. На практиці в системах реального часу існує необхідність отримувати, опрацьовувати та відображати в різних формах великі обсяги даних у реальному часі їх надходження. Також в складних швидкоплинних або тривалих процесах фізичних об’єктів у різних масштабах реального часу виникає потреба у проведенні всебічного, багатоаспектного опрацювання та відображення отриманих даних. Як правило, ці дані мають просторово-часовий характер, що значно ускладнює завдання (рис. 1). Для розв’язання такого класу задач необхідно розробляти програмні засоби, які здатні не тільки швидко обробляти велику кількість даних, а і візуалізовувати їх без значних затрат апаратних ресурсів. З розвитком комп’ютерної техніки вирішення цієї проблеми стало можливим.

Аналіз існуючих програмних пакетів відображення даних

Більшість існуючих програмних засобів передбачають роботу зі статичними даними або просторово-часовими зрізами даних, що значно обмежує можливості аналізу і відтворення необхідної інформації.

Найпопулярнішими сьогодні програмами візуалізації дво- і тривимірних даних є Grapher і Surfer, розроблені американською фірмою Golden Software. Для прикладу, програма Surfer дає змогу відображати поверхню розміром 10000 х 10000 вузлів.

Вказані програмні засоби призначені для побудови графіків і поверхонь за готовими даними або відомими залежностями. Їх перевагами є широкий набір методів інтерполяції дискретних функцій та можливість налаштування потрібних параметрів в запропонованих методах. Головний їх недолік – здатність відображати дані лише для однієї поверхні і відсутність функції відтворення просторових залежностей отриманих величин в часі.

Крім основної функції візуалізації просторових даних, програмний пакет Surfer дає змогу [1]:

– розраховувати об’єм між двома поверхнями;

– замінити одну регулярну сітку на іншу;

– провести перетворення поверхонь за допомогою математичних операцій з матрицями;

– здійснити розсікання поверхні (розрахунок профілю);

– визначити площу поверхні;

– провести згладжування поверхонь з використанням матричних або сплайн-методів;

– здійснити перетворення форматів файлів.

Постановка задачі

Оскільки сьогодні не існує програм, що відтворюють поведінку поверхні в часі, було поставлено задачу розробити саме такий програмний пакет. Крім того, він повинен мати такі можливості:

– перегляду поверхні під будь яким кутом;

– масштабування поверхні для кращого відображення як усієї просторової залежності, так і окремих її ділянок;

– визначення ділянок, в яких поверхня набуває заданих значень, і виділення цих ділянок вибраним кольором за побажаннями користувача;

– відтворення лише деякої частини поверхні, заданої необхідними вузлами.

– керування часовим відтворенням просторових поверхонь.

Опис програмного пакета

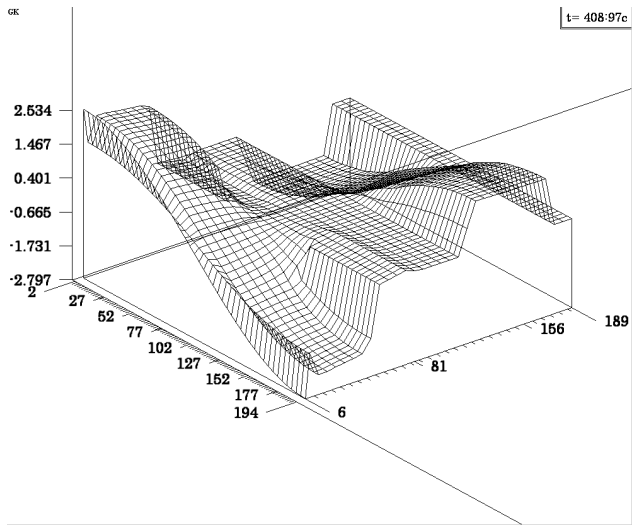

Рис. 1. Приклад відображення просторово-часової поверхні

У програмі Surfer сітка поверхні, що опрацьовується, генерується відразу після опрацювання даних. Потім програма зберігає її в оперативній пам’яті у вигляді цифрового малюнка, що значно знижує затрату ресурсів процесора. Цифрове зображення оновлюється тільки якщо користувач змінив параметри зображення.

Відображати поверхню в часі необхідно за іншим принципом, оскільки сам процес відображення нагадує відтворення відеофайла. Програма повинна генерувати сітку поверхні не менше ніж 24 рази в секунду для плавного передавання зміни динаміки зображень. У нашому випадку не можна застосовувати методи, використані в програмі Surfer, крім того, основною проблемою, яка виникає під час розроблення заданої програми, стає її швидкодія.

Інтерфейс програми розміщено в стандартному вікні Windows, що забезпечує звичний для пересічного користувача інтерфейс і дає змогу значно прискорити розроблення за допомогою стандартного інструментарію для написання програм під Windows. Але стандартні компоненти для побудови графіків і поверхонь, які інтегруються в програми під Windows, є дорогими з погляду ресурсів і не є ефективними для виконання поставленої задачі. Ще один їх недолік – це мерехтіння зображення, яке можна зауважити в графічних редакторах, коли користувач переміщує виділену частину зображення.

У результаті було прийнято рішення використати ігровий двовимірний рушій Haafs Game Engine (HGE), який легко під'єднати до програм під Windows. Він дає змогу без значних затрат ресурсів комп'ютера виконувати різноманітні графічні функції, зокрема виводити зображення для декількох тисяч ліній. Однією з переваг його застосування є відсутність мерехтіння зображення при перемальовуванні. Також він простий у використанні і підтримується всіма версіями Windows, починаючи з Windows 98, навіть Windows Vista, яка не підтримувала багатьох програм, що працювали з попередніми версіями.

HGE – центральний компонент програми, який виводить зображення поверхні і перетворює її. Він відповідає за всі технічні аспекти і значно полегшує розроблення за рахунок уніфікації та систематизації її внутрішньої структури [5].

HGE може працювати в двох режимах: у звичайному і як дочірнє вікно. Програма в звичайному режимі активізується в окремому вікні, яке за потреби може розгортатися на весь екран. Проблема полягає в тому, що до такого вікна не можна додати стандартних компонентів Windows (кнопок, слайдерів та ін.). Щоб використати HGE у звичайному вікні Windows і тим самим отримати змогу додавати до цього вікна необхідні стандартні компоненти, необхідно використовувати HGE в режимі дочірнього вікна. Але в такому режимі кожного разу вручну потрібно активізувати рушій з певним періодом, використовуючи компоненту «таймер». У звичайному режимі цей рушій треба запустити лише раз. Далі він працює самостійно, виконуючи по колу задані функції, поки програма не дасть команду завершити роботу. В режимі дочірнього вікна його треба активізовувати безперервно і при кожному запуску він завершує роботу, виконуючи задані функції лише один раз. Причиною цього є особливості побудови програм під Windows.

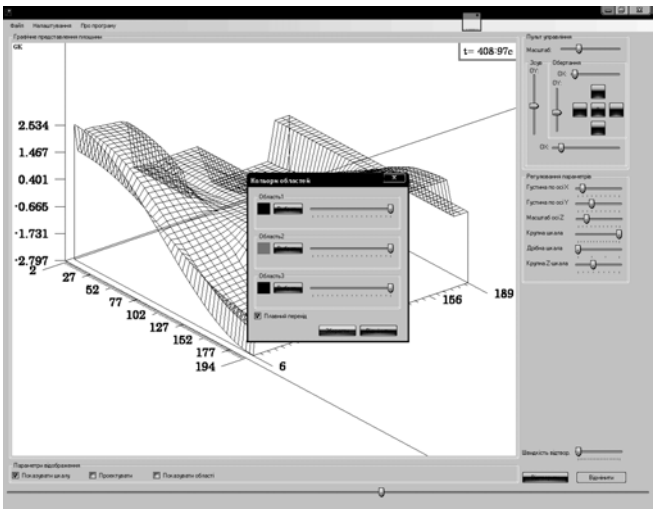

Рис. 2. Інтерфейс програмного продукту візуалізації просторово-часових поверхонь

Інтерфейс розробленого програмного продукту нагадує інтерфейс відеоплеєра з набором додаткових можливостей (рис. 2). Користувач може зупинити відтворення, "перемотати" на будь-який момент часу, відкрити іншу групу файлів, а також у реальному часі змінювати положення поверхні в просторі, швидкість відтворення та інші параметри.

Поверхня відображається в програмі аналогічно до інших подібних програм, тобто у вигляді сітки. Кожен вузол сітки є об'єктом класу «вузол» і містить шість змінних: три просторові координати (x, y, z), дві координати проекції вузла на площину (x_s, y_s), тобто координати у вікні, де відтворюється поверхня і адреса комірки пам'яті, що містить колір, вибраний користувачем.

Три просторові координати – це значення, отримані на основі даних з файлів, координати по x і y та саме значення функції у вузлі. Дві координати проекції – це координати вузла, спроектованого на площину. Використання координат проекції при побудові зображень значно зменшує завантаження процесора. Ці координати розраховуються лише за необхідності (коли користувач повернув поверхню в просторі або поверхня відображається в часі, тому змінюється

значення вузла по осі z). Якби цих змінних не було, нам необхідно було б кожного разу проектувати всі вузли на площину, навіть коли користувач просто змінив колір або розмір шрифту поділок. По суті, свідомо надлишково використовуються ресурси оперативної пам'яті для збільшення швидкодії програми. Крім того, якщо зсунути проекцію поверхні горизонтально, змінюється координата проекції x_s, а координата y_s залишається незмінною. Тому для вузлів розраховують тільки координати x_s, що значно економить ресурси процесора.

Додатковою функцією розробленого програмного продукту є відслідковування областей, в які потрапляють точки поверхні. Наприклад, поставлено задачу виділити вузли, в яких значення функції не перевищує 18, в яких значення функції в межах 18 до 24 і в яких значення перевищує 24. Якщо наша поверхня – це температурний зріз у просторі, за допомогою цієї функції легко можна виокремити зони температурних режимів. Значення і кольори цих зон вибираються в налаштуваннях.

За замовчуванням поверхня виводиться чорним кольором. Інформація про кольори зберігається стосовно кожного вузла і змінюється лише в режимі відтворення або коли користувач змінив значення областей.

Коли будується поверхня, координати проекцій вузлів на площину з'єднуються лініями певного кольору. Колір лінії залежить від значень кольору у з'єднуваних вузлах. Значення кольору вузла залежить від області, в межах якої він знаходиться. Залежно від опції «плавний перехід» у налаштуваннях кольору областей, лінія між вузлами з різними значеннями кольорів відтворюється або проміжним кольором, або не відтворюється взагалі.

Чому в об'єкті вузла міститься адреса комірки з кольором, а не значення кольору? Припустімо, користувач замінив колір певної області площини з зеленого на синій. Якби в об'єкті містились значення кольору, довелося б переприсвоювати ці значення в усіх вузлах зеленого кольору. У зв'язку з цим просто змінюють значення в комірці зі сталою адресою, використовуючи лише одне присвоєння.

Робота з поверхнею розпочинається тоді, коли користувач вибирає групу файлів з даними про поверхню. Файли з даними структуровані певним чином. Значення, записане на самому початку файла – це момент часу, якому відповідають дані. Далі в три колонки виводяться дані за вузлами: координата по осі x, координата по осі y і саме значення функції (тобто координата по осі z). Перші дві колонки значень впорядковані однаково для всіх файлів. Тому спочатку програма впорядковує файли за часом, вказаним на початку, зчитує координати вузлів з першого файла і присвоює початкові значення по осі z. Опрацьовуючи дані з наступних файлів, програма нехтує координатами вузлів, записуючи лише значення в третій колонці і розраховує приріст значень у вузлах. Приріст – це швидкість, з якою значення в певному вузлі змінюється в часі. Приріст може бути як додатний, так і від'ємний.

Дані з файлів заносяться в об'єкти класу «зберігач даних». Кожен з таких об'єктів містить значення усіх вузлів в момент часу, вказаного на початку файла, саме значення часу і прирости вузлів. Значення приросту дорівнює різниці значення вузла з поточного і попереднього файлів, поділеної на різницю часу, вказаного в попередньому і поточному файлах. Коли дані з файлів заносяться в пам'ять, програма створює об'єкт класу «зберігач даних» для кожного файла. Потім ці об'єкти сортуються за часом і так розбивається період між початком і кінцем вибірки.

Приріст потрібен для того, щоб розрахувати значення вузла у будь-який момент часу. Під час відтворення користувач може змінювати швидкість відображення поверхні або "перемотати" на інший довільний момент часу. Тому першим параметром функції, яка перетворює поверхню, є момент часу. Залежно від дій користувача програма розраховує момент часу, в який необхідно показати поверхню, і передає його у функцію перетворення. Функція перетворення спочатку визначає, в якому з відрізків часу знаходиться цей момент, заданий як параметр, а потім на основі значення на кінці цього відрізка, різниці часу і приросту визначає поточне значення вузла.

HGE – двовимірний рушій, який не призначений для побудови поверхонь. Для того, щоб за його допомогою візуалізувати тривимірне зображення, необхідно щоразу перетворювати зображення на двовимірне.

Перетворення тривимірних координат на двовимірні враховує кути нахилу і повороту зображення (відповідно α і β).

Розрахунок двовимірної координати за віссю 0-*x*:

$$x_s = (y \cdot \cos(\beta) + x \cdot \sin(\beta)) \cdot z_{plus} + x_{plus}. \quad (1)$$

Розрахунок двовимірної координати за віссю 0-*y*:

$$y_s = (z \cdot \cos(\alpha) + (y \cdot \sin(\beta) - x \cdot \cos(\beta)) \cdot \sin(\alpha)) \cdot z_{plus} + y_{plus}, \quad (2)$$

де α і β – кути нахилу і повороту; *x*, *y*, *z* – координат вузла в просторі; x_{plus} та y_{plus} – значення зсуву поверхні на площині (з їх допомогою можна зсувати поверхню вертикально і горизонтально); z_{plus} – коефіцієнт розміру поверхні (з його допомогою можна збільшувати і зменшувати поверхню за віссю *z*).

Оскільки для кожного вузла значення кутів однакові, то однаковими будуть і значення їх синусів та косинусів. Тому cos(β), sin(β), cos(α) та sin(α) вираховуються лише на початку і слугують параметрами для функцій перетворення.

З формули розрахунку координати x_s можна зауважити, що вона залежить лише від кута повороту зображення. Тобто, як би не нахилялась поверхня відносно осі *z*, двовимірна координата x_s залишається незмінною. Аналогічно при зсувах поверхні змінюється лише одна з двовимірних координат. Тому при маніпуляціях над поверхнею здійснюються лише певні необхідні перетворення, що значно зменшує завантаження процесора.

Вікно з поверхнею оновлюється тільки тоді, коли це потрібно, за допомогою стандартної компоненти Windows під назвою “timer”. Вона створює окремий потік (thread), в якому з заданим періодом виконується функція оновлення вікна з поверхнею. Період таймера задається в мілісекундах, і для нашого таймера становить 42, що наближено дорівнює 1/24 секунди (тобто часу перебування на екрані одного кадру).

Висновок

На основі ігрового двовимірного рушія Haafs Game Engine було розроблено одночасно простий і ефективний програмний продукт, який дає змогу відтворювати з необхідними параметрами візуалізації зміну просторових даних поверхні в часі.

1. Колесов А., Павлова О. Пакет Surfer – обработка и визуализация двумерных функций // КомпьютерПресс. – 1999. – № 2. 2. Ильин В.А., Позняк Э.Г. Аналитическая геометрия. – М.: ФИЗМАТЛИТ, 2002. – 240 с. 3. David Hilbert; Cohn-Vossen, S. Geometry and Imagination. American Mathematical Society, 1999. 4. Bredon, Glen E. Topology and Geometry. Springer-Verlag, 1993. 5. http://uk.wikipedia.org/wiki/Ігровий_рушій.