УДК 004.4

В.І. Межуєв

Бердянський державний педагогічний університет

ВІДНОВЛЕННЯ ТРИВИМІРНИХ ОБ'ЄКТІВ ЗА ЇХ СЛІДАМИ НА СИСТЕМІ ВЗАЄМНО ПЕРПЕНДИКУЛЯРНИХ ПЛОЩИН ІЗ ВИКОРИСТАННЯМ ІНТЕРФЛЕТАЦІЇ ФУНКЦІЙ

У статті висвітлюється метод відновлення структури тривимірних об'єктів за їх слідами на системі взаємно перпендикулярних площин. Метод використовує оператори інтерполяції, побудовані на основі інтерфлетації функцій. Наводяться результати практичного застосування методу з метою реконструкції тривимірного зображення людського мозку за системою його паралельних перетинів.

відновлення тривимірних об'єктів; інтерфлетація функцій; тривимірні перетворення

1. Постановка проблеми

У практиці дослідження томографічних зображень часто виникає задача за відомою сукупністю перетинів отримати зображення тіла у довільному перетині [1]. Така задача, наприклад, виникає у медицині та біології при дослідженні кори головного мозку людини або ж піддослідних тварин. Як правило, інформація про властивості тіла у даних перетинах отримується у вигляді фотографічних зобра-

жень, які поступають у пам'ять комп'ютера з томографа у растровому форматі (тобто файлів з розширенням *bmp* – від англ. *bitmap* – бітова карта).

Растровий формат томографічних зображень є зручним для математичної обробки, у даному випадку кожній точці дисплею ЕОМ з координатами *(x, y)* ставиться у відповідність функція інтенсивності кольорового зображення за тою або іншою моделлю кольору (наприклад, *RGB*, *CMYB*, *HSB* моделями).

У даній роботі пропонується метод отримання розподілу деякої фізичної характеристики $u(x,y,z)$ внутрішньої структури тривимірного тіла (наприклад, його щільності) у довільному перетині. Тіло ми будемо вважати без обмеження загальності метода повністю розміщеним в одиничному кубі $D=[0,1]^3$.

Джерелом інформації про функцію $u(x,y,z)$, тобто внутрішню структуру тіла, будемо вважати її сліди на системі взаємно перпендикулярних площин:

$$x=i/M_1, i=\overline{0,M_1};\quad y=j/M_2, j=\overline{0,M_2};$$
$$z=k/M_3, k=\overline{0,M_3},$$

тобто набір функцій

$$u(i/M_1,y,z),\ i=\overline{0,M_1};$$
$$u(x,j/M_2,z),\ j=\overline{0,M_2};\ u(x,y,k/M_3),\ k=\overline{0,M_3}.$$

Для відновлення внутрішньої структури тіла ми використовуємо оператори сплайн-інтерполяції O_1, O_2, O_3 функції u за змінними x,y,z відповідно:

$$O_1u(x,y,z)=\sum_{i=0}^{M_1}Sp_{M_1,i}(x)u(i/M_1,y,z);$$
$$O_2u(x,y,z)=\sum_{j=0}^{M_2}Sp_{M_2,j}(y)u(x,j/M_2,z);\qquad (1)$$
$$O_3u(x,y,z)=\sum_{k=0}^{M_3}Sp_{M_3,k}(z)u(x,y,k/M_3),$$

де $Sp_{M_1,i}(x)$ – базисні сплайни степеня m (m = 1, 2, 3) із такими властивостями:

$$Sp_{M_1,i}(p/M_1)=\delta_{i,p},\quad i,p=\overline{0,M_1}.$$

Аналогічно визначаються також сплайни $Sp_{M_2,j}$, $Sp_{M_3,k}$.

Інтерлінація функцій багатьох змінних – це відновлення цих функцій за допомогою їх слідів і слідів їх часткових похідних (або деяких інших диференціальних операторів) даного порядку на системі ліній. Інтерфлетація функцій багатьох змінних – це відновлення цих функцій за допомогою їхніх слідів і слідів їхніх часткових похідних (або деяких інших диференціальних оператори) даного порядку на системі поверхонь (в окремому випадку на площинах). Інтерлінація і інтерфлетація – природне узагальнення інтерполяції функцій багатьох змінних [2].

Оператор сплайн-інтерфлетації

$$Lu(x,y,z)=\begin{pmatrix}O_1+O_2+O_3-O_1O_2-\\ O_1O_2-O_2O_3+O_1O_2O_3\end{pmatrix}u(x,y,z)\quad (2)$$

має наступні властивості

$$Lu(i/M_1,y,z)=u(i/M_1,y,z);\ i=\overline{0,M_1};$$
$$Lu(x,j/M_2,z)=u(x,j/M_2,z);\ j=\overline{0,M_2};\qquad (3)$$
$$Lu(x,y,k/M_3)=u(x,y,k/M_3);\ k=\overline{0,M_3}.$$

Доведення (3) наводиться у [2].

У реальному випадку, якщо експериментальні дані задані з похибкою (що як раз і має місце у даній роботі) виконується наступна теорема.

Теорема. Якщо експериментальні дані

$$\tilde u(i/M_1,y,z),\ i=\overline{0,M_1};$$
$$\tilde u(x,j/M_2,z),\ j=\overline{0,M_2};\ \tilde u(x,y,k/M_3),\ k=\overline{0,M_3}$$

задані із максимальною похибкою ε, а ψ – випадкова величина рівномірно розподілена в області D тобто

$$\tilde u_{1,i}(y,z)=u(i/M_1,y,z)+\varepsilon\psi,\ i=\overline{0,M_1};$$
$$\tilde u_{2,j}(x,z)=u(x,j/M_2,z)+\varepsilon\psi,\ j=\overline{0,M_2};$$
$$\tilde u_{3,k}(x,y)=u(x,y,k/M_3)+\varepsilon\psi,\ k=\overline{0,M_3},$$

то оператор

$$L\tilde u(x,y,z)=$$
$$=\begin{pmatrix}O_1+O_2+O_3-O_1O_2-O_1O_2\\ -O_2O_3+O_1O_2O_3\end{pmatrix}\tilde u(x,y,z)$$

буде мати похибку

$$Ru(x,y,z)=u(x,y,z)-L\tilde u(x,y,z),$$

яка задовольняє наступному співвідношенню:

$$\|Ru\|_{C(D)}=O\left(M_1^{-m-1}M_2^{-m-2}M_3^{-m-3}\right)+O(\varepsilon).$$

Доведення.

$$Ru(x,y,z)=u(x,y,z)-L\tilde u(x,y,z)=$$
$$=u(x,y,z)-L(u(x,y,z)+\varepsilon\psi(x,y,z))=$$
$$=u(x,y,z)-L(u(x,y,z))-\varepsilon\ L\psi(x,y,z).$$

З цієї тотожності випливає нерівність $\|Ru\|_{C(D)}\le\|u-Lu\|_{C(D)}+\varepsilon\ \|L\psi\|_{C(D)}$ з якої і отримується доведення.

2. Практична реалізація

Практична реалізація метода сплайн-інтерфлетації була здійснена нами на мові *C++* у системі візуального програмування *Builder 6.0* з метою реконструкції тривимірного зображення людського мозку за системою його паралельних перетинів. Програма *Brain Reconstruction* працює в операційних середовищах Windows 98, 2000, XP.

Зовнішній вигляд та інтерфейс програми *Brain Reconstruction* зображений на рис. 1.

Вхідними даними програми *Brain Reconstruction* є сукупність рисунків (файлів у растровому форматі *bmp*), що зображають перетини тіла у системі взаємно перпендикулярних площин

$$x=x_i, y=y_j, z=z_k,\ i=\overline{1..M_1}, j=\overline{1..M_2}, k=\overline{1..M_3}.$$

За цією сукупністю файлів будується тривимірний масив рисунків *Pictures*, причому ознакою на-

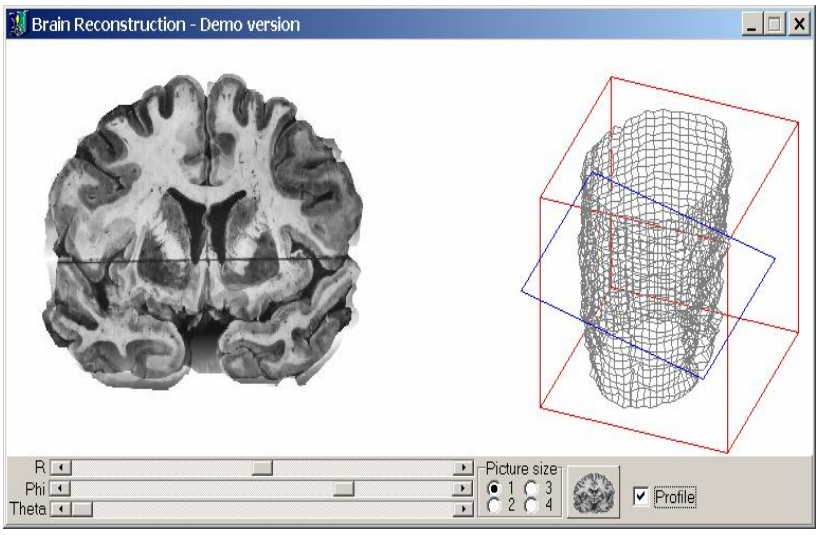

Рис. 1. Зовнішній вигляд та інтерфейс програми *Brain Reconstruction*

лежності файла до однієї з трьох систем паралельних перетинів є його назва (наприклад, x1.bmp, x2.bmp, x3.bmp – рисунки, що зображають перетини тіла у перпендикулярній осі OX площині).

Запропонований метод є загальним та дозволяє відновлювати структуру тривимірного тіла при різних довільних відстанях між паралельними перетинами. Але з метою спрощення алгоритму та максимально наближення до практики сканування томографічних зображень (технічно простіше отримати перетини із сталим кроком) вважається, що відстань між перетинами є однаковою для будь-якої із систем площин. Вхідні дані повинні знаходитися у той же папці, де розташований файл *Brain.exe.* Програма зчитує всі файли рисунків, що відповідають наступним критеріям:

1. Назва файла повинна починатися з латинських букв x, y або z, що відбиває належність рисунка до однієї з систем взаємно перпендикулярних площин.
2. Наступними символами повинні бути арабські цифри, що слугують для визначення порядкового номера рисунка у відповідній системі площин.

Всі знайдені коректні файли (тобто файли, що відповідають стандарту BMP) заносяться у тривимірний масив рисунків, а їх кількість у відповідні змінні M_1, M_2, M_3.

Оператори сплайн-інтерфлетації реалізовані у програмі у виді окремих функцій O1, O2, O3, O12, O13, O23, O123, параметрами яких є змінні *x, y, z* підвищеної точності (типу *long double* мови C++).

Функції O12, O13, O23, O123 використовують значення, що повертають операторні функції O1, O2, O3. Кожна із функцій O1, O2, O3 здійснює обчислення номерів найближчих до точки із координатами x, y, z шарів i, j, k та за цими номерами звертається до конкретних елементів тривимірного масиву рисунків. Наприклад, якщо обчислений номер шару i, то звертання до елементу тривимірного масиву *Pictures* здійснюється у *C++ Builder* наступним чином:

Pictures[1][i]->Canvas->Pixels[x][y].

Розкриємо сутність наведеного виразу. Кожна форма *C++ Builder* має полотно (англ. *canvas*) для рисування, що, у сутності, є двовимірним масивом пікселів (від англ. *picture elements* – елемент зображення) *Pixels[m][n]*, де m, n – ширина та висота робочої поверхні форми.

Кожному елементу цього масиву, в свою чергу, поставлений у відповідність атрибут, тобто колір. Відзначимо, що при програмуванні у *C++ Builder* використовується *RGB* модель кольору, відповідно до якої будь який колір можна зобразити як суперпозицію трьох базових кольорів – червоного (*Red*), зеленого (*Green*) та синього (*Blue*).

Кожна з компонент кольору – *Red, Green* або *Blue* задається у *C++ Builder* одним байтом, тобто може відбивати $2^8 = 256$ відтінків кольору (значення інтенсивності змінюється від 0 до 255, число 255 відбиває колір найбільшої інтенсивності). Кількість кольорів, що можна задати трьома байтами складає $256^3 = 16\,777\,216$, що близько до кольорової здатності людського ока (ця модель має назву *True Color*).

Для подальшого викладу зручно перейти у шістнадцятирічну систему числення, ознакою якої на мові C++ є комбінація символів 0x. Повністю установлені байти (FF) свідчать про найбільшу інтенсивність відповідної компоненти кольору (табл. 1).

Таблиця 1

Значення байт кольорів у моделі RGB

Колір	**1 байт**	**2 байт**	**3 байт**	**Значення**
Червоний	00	00	FF	0x0000FF
Зелений	00	FF	00	0x00FF00
Синій	FF	00	00	0xFF0000

Таким чином, 0xFFFFFF є значенням кольору із найбільшою інтенсивністю (тобто білим кольором), 0x000000 є значенням чорного кольору. Діапазон між 0x000000 та 0xFFFFFF з рівними значеннями кожного з компонентів кольору дозволяє оперувати відтінками сірого (наприклад, 0x111111 або 0xAAAAAA). Особливість комп'ютерної реалізації операторів O_1, O_2, O_3 полягає у необхідності дії на кожну з RGB компонент кольору окремо. Таким чином, у функціях O1, O2, O3 здійснюється виділення окремих компонент кольору в даній точці. Наприклад, вираз

Pictures[3][k]->Canvas->Pixels[x][y]&0x0000FF

здійснює виділення червоної компоненти кольору з пікселя, що відноситься до перпендикулярної осі *OZ k-ї* площини та має екранні координати x, y. Для виділення окремого байту (компоненти кольору) використовується арифметична операція мови *C++* "і" (&), що обнуляє значення всіх інших байтів, крім

заданого другим операндом (у нашому прикладі це 0x000000FF). Для пояснення дії цієї операції перейдемо у двійкову систему числення. Наприклад, ми маємо наступне значення кольору у двійковому форматі 101010101010101010101010. Відзначимо, що $000000FF_{16} = 000000000000000011111111_2$ (кожен байт задається 8 двійковими цифрами, тобто для завдання кольору як сукупності трьох компонентів використовуються загалом 24 розряди). Завдання окремої цифри у шістнадцятирічному форматі потребує 4 біта.

Здійснимо арифметичну операцію і (&) між двома байтами:

101010101010101010101010
<u>000000000000000011111111</u>
000000000000000**10101010**

Таким чином, здійснюється виділення компоненти кольору (у даному випадку крайнього правого байту – червоної компоненти).

Оперування з іншими компонентами кольору потребує їх зсуву на місце крайнього правого байту, для чого використовується бітова операція зсуву мови *C++* “>>”. Наприклад, для зеленої компоненти кольору маємо

(Pictures[3][k]->Canvas->
Pixels[x][y]&0x0000FF00)>>8,

де останнім операндом операції >> є кількість біт, на які здійснюється зсув значення.

Після проведення підрахунків із окремих компонентів знову формується значення кольору, для чого відповідні байти зсуваються на своє місце:

Color = R + (G<<8) + (B<<16).

3. Реалізація тривимірної графіки

Задача побудови перспективного зображення зводиться до перетворення сукупності координат реального об'єкта в екранні координати. Цей процес здійснюється в два етапи:

1. Реальні координати точки (x, y, z) за допомогою *видового перетворення* трансформуються у видові координати (x_v, y_v, z_v).
2. Видові координати (x_v, y_v, z_v) за допомогою *перспективного перетворення* трансформуються в екранні координати (x_e, y_e).

Для виконання видових перетворень задаються місцеположення всіх точок об'єкта у світовій системі координат та точка спостереження, що збігає з оком спостерігача.

Для того, щоб одержати зображення тривимірного об'єкта на екрані ЕОМ потрібно спроектувати його на площину. З цією метою вводиться видова система координат, в якій об'єкт проектується на площину $X_v0_vY_v$, а вісь Z_e0_e визначає напрямок спостереження (рис. 2).

Точку спостереження зручно визначати в сферичній системі координат за такими параметрами: θ – кут, що складає з віссю OX проекція лінії спостереження на площину XOY; φ – полярний кут, що складає лінія спостереження з віссю OZ; ρ – відстань до точки спостереження. Саме трійка цих величин (θ, φ, ρ) є вхідними параметрами інтерфейсу

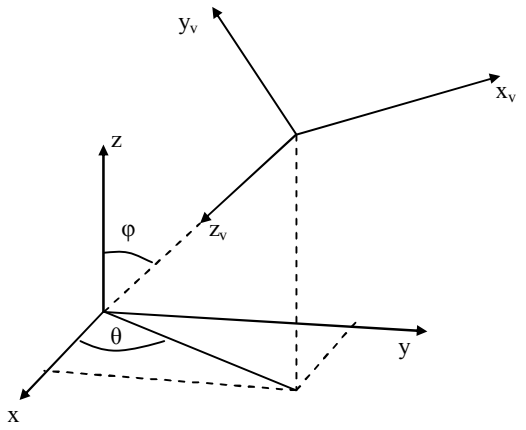

Рис. 2. Визначення точки спостереження у тривимірному просторі

програми *Brain Reconstruction* (рис. 1). За цими даними програма здійснює видові перетворення та проектує зображення зрізу мозку на екран ЕОМ.

Нехай положення точки в тривимірному просторі задається координатами (x, y, z). Тоді видове перетворення можна записати у формі:

$$|x_v\ y_v\ z_v\ 1| = |x\ y\ z\ 1|\cdot V,$$

де $V = \begin{vmatrix} -\sin\theta & -\cos\varphi\cos\theta & -\sin\varphi\cos\theta & 0 \\ \cos\theta & -\cos\varphi\sin\theta & -\sin\varphi\sin\theta & 0 \\ 0 & \sin\varphi & -\cos\varphi & 0 \\ 0 & 0 & \rho & 1 \end{vmatrix}$.

Координати x_v і y_v точки об’єкта можна використовувати для побудови зображення на екрані, у цьому випадку буде отримана *ортогональна* проекція, коли кожна точка об'єкта P проектується в точку P′ проведенням лінії, перпендикулярної площини XOY (або ж, іншими словами, паралельній лінії спостереження). Така проекція має місце при віддаленні точки спостереження в нескінченність, при цьому рівнобіжні лінії тривимірного об'єкта залишаються рівнобіжними і на його проекції.

Висновки

В даній роботі висвітлюється метод відновлення структури тривимірних об’єктів за їх слідами на системі взаємно перпендикулярних площин, що використовує оператори інтерполяції, побудовані на основі інтерфлетації функцій. Практично даний метод був використаний з метою реконструкції тривимірного зображення людського мозку за системою його паралельних перетинів.

Список літератури

1. Крейн С.Г., Петунин Ю.И., Семенов Е.М. Интерполяция линейных операторов. – М.: Наука, 1978. – 400 с.

2. Литвин О.М. Інтерлінація функцій та деякі її застосування. – Х.: Основа, 2002. – 544 с.

3. Наттерер Ф. Математические аспекты компьютерной томографии. – М.: Мир, 1990. – 286 с.

Надійшла до редколегії 15 01.2007

Рецензент: д-р фіз.-мат. наук, проф. В.В.Кідалов, Бердянський державний педагогічний університет, Бердянськ.