ДЕРЖАВНИЙ УНІВЕРСИТЕТ «Житомирська політехніка»

Кафедра метрології та інформаційно-вимірювальної техніки

КУРСОВА РОБОТА

з дисципліни

« Інженерна та комп’ютерна графіка»

«Розробка методів геометричних перетворень об’єктів»

Студента (ки) _2_ курсу АТ-29_ групи
напряму підготовки__________________
*спеціальності*__*АТ*___ ____________
_______*Кузнецова К.Ю.*_____________
(прізвище та ініціали)
Керівник __*д.т.н, проф.*___ ______
_________*Подчашинський Ю.О.*______
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _________________
Кількість балів: __________Оцінка: ECTS _____

Члени комісії
________________ ______________________
(підпис) (прізвище та ініціали)
________________ ______________________
(підпис) (прізвище та ініціали)
________________ ______________________
(підпис) (прізвище та ініціали

м. Житомир 2020 рік

Зміст

					ММАТ.460. 007.007 – КР
Змн.	*Арк.*	*№ докум.*	*Підпис*	*Дата*	
Розроб.		*Кузнецова*			*Комп'ютерна графіка*
Перевір.		*Подчашинський*			
Реценз.					*Пояснювальна записка*
Н. Контр.					
Затверд.					

Літ.	*Арк.*	*Акрушів*
	2	38

ДУ «Житомирьска політехніка», АТ-29

ЗАВДАННЯ КУРСОВОЇ РОБОТИ

Номер варіанту	Розмір зображення		Двовимірна геометрична фігура – прямокутник			Зображення літер
	ширина *N*, дискр. точок	висота *M*, дискр. точок	ширина *H*, диск. точок	висота *L*, диск. точок	кут повороту *alfa*, градусів відносно *Ox* проти год. стрілки	
7	250	400	80	80	60	ЄІЇ

Номер варіанту	Параметри геометричного перетворення								
	Зсув, дискр. точок			Масштабування			Кут повороту, градусів проти год. стрілкі відновно вказаної осі		
	T_x	T_y	T_z	S_x	S_y	S_z	$Teta_x$	$Teta_y$	$Teta_z$
7	30	10	0	1,2	1,5	1,5	25	40	0

ПОСЛІДОВНІСТЬ ВИКОНАННЯ КУРСОВОЇ РОБОТИ

1. Для двовимірного зображення геометричної фігури згідно варіанту завдання дослідити задане геометричне перетворення.

1.1. Сформувати двовимірне зображення заданої геометричної фігури

1.2. Виділити 3 контрольні точки геометричної фігури та визначити їх координати:

а) шляхом розрахунку

б) шляхом вимірювань на сформованому зображенні з п.1.1. impixel

1.3. Обчислити координати контрольних точок після застосування до сформованого зображення заданого геометричного перетворення

1.4. Визначити матрицю та операцію геометричного перетворення в MATLAB та отримати зображення-результат перетворення

1.5. Визначити на зображенні-результаті перетворення координати контрольних точок, порівняти їх з результатами п.1.3.

1.6. Виконати в MATLAB зворотне геометричне перетворення шляхом:

а) застосування матриці оберненого перетворення відносно результатів п.1.4.

б) застосування геометричного перетворення на основі координат початкових та результуючих контрольних точок з п.1.5 та п. 1.2.

ВСТУП

В даний час комп'ютерна графіка являє собою спеціальну область інформатики, що займається методами і засобами створення, перетворення, обробки, зберігання і виведення на друк зображень за допомогою цифрових обчислювальних комплексів, називають.

Розвиток комп'ютерної графіки як самостійної галузі інформатики почалося в дев'яностих роках минулого століття. Цьому сприяло, з одного боку, різке підвищення технічних характеристик (ємність запам'ятовуючих пристроїв, швидкодія і розрядність процесора, можливість роботи з масивами чисел, представлених у формі з плаваючою комою) і зниження вартості апаратного забезпечення, з іншого боку, поява адаптованого до роботи з графікою як базового, так і прикладного програмного забезпечення.

За останнє десятиліття діапазон застосування комп'ютерної графіки істотно розширився. Наприклад, раніше її могли використовувати тільки фахівці, що працюють в деяких предметних областях:

- в медицині (комп'ютерна томографія);
- в наукових дослідженнях (астрономія, мікробіологія);
- в інженерно-технічних розробках (проектування будівель, споруд, літальних апаратів).

Комп'ютерна графіка перетворилася з вузькоспеціальної області інтересів кількох професій у справу, якого прагнуть присвятити себе безліч людей.

Створено різноманітне апаратне і програмне забезпечення для отримання зображень самого різного виду та призначення - від простих креслень до реалістичних образів природних об'єктів.

Кінцевим результатом застосування засобів комп'ютерної графіки є зображення, яке може використовуватися для різних цілей.

Реалістичність сприйняття людиною складного комп'ютерного зображення визначається вмінням розробника математичної моделі зображуваного об'єкта або процесу достовірно повторити на екрані його розвиток в просторі і в часі. Модель включає в себе систему рівнянь і алгоритмів їх реалізації.

Математичної основою побудови моделі є рівняння, що описують форму і рух об'єктів. Все різноманіття геометричних об'єктів є комбінацією різних примітивів – найпростіших фігур, які в свою чергу складаються з графічних елементів – точок , ліній і поверхонь.

Під комп'ютерною геометрією розуміють математичний апарат, застосовуваний у комп'ютерній графіці.

Комп'ютерна геометрія є математичний апарат, покладений в основу комп'ютерної графіки. У свою чергу, основою комп'ютерної геометрії складають різні перетворення точок і ліній. При використанні машинної графіки можна за бажанням змінювати масштаб зображення , обертати його, зміщувати і трансформувати для поліпшення наочності перспективного зображення. Всі ці перетворення можна виконати на основі математичних методів .

Перетворення, як і комп'ютерну геометрію, поділяють на двовимірні, або перетворення на площині, і тривимірні або просторові .

Але, з точки зору, нашої спеціальності розглянемо комп’ютерну графіку в інженерно-технічних розробках.

Наприклад, потрібно спостерігати з космосу за певними об’єктами, це в свою чергу передбачає наявність дискретного за часом спостереження з невеликим тимчасовим інтервалом, і тому, коли рухома камера фіксує напівтоновий образ об'єкта, що спостерігається (оптичну поверхню) у вигляді послідовності зображень, то цей образ від знімка до знімка деформується внаслідок перспективних спотворень і зміни положення камери. Геометрія відповідних деформацій моделюється проективними перетвореннями, які становлять більш великий клас, ніж відомі перетворення евклідової геометрії. Відновлення просторового рельєфу по стереоснимку призводить до проблеми точного встановлення координати відповідності елементів стереозображень. Для усунення цієї проблеми потрібно виділити пари реперних фрагментів і оцінювати параметри «розбіжності» відповідних точок, за якими можна відновити функцію геометричного перетворення і оцінити поверхню тривимірної сцени (рельєф).

Отже, застосування комп’ютерної графіки в інженерних цілях, передбачають оперативної взаємодії людини і комп'ютера, а також задачі числової обробки, розшифровки і передачі зображень.

1 ОГЛЯД МЕТОДІВ ГЕОМЕТРИЧНИХ ПЕРЕТВОРЕНЬ

Геометричні об'єкти на площині й у просторі можна піддавати ряду різних перетворень. Перетворення включає в себе обртання, масштабування, відображення. Розглянемо кожне з цих перетворень окремо.

1.1 Обертання

Розглянемо площину трикутника *ABC* (рис.1.1) і за допомогою наступного перетворення повернемо його на 90° проти годинникової стрілки відносно початку координат

$$\left[\grave{O}\right] = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

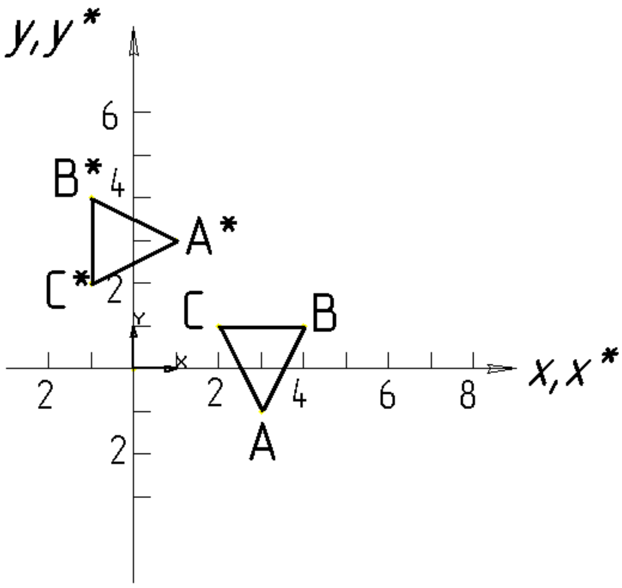

Рис.1.1

Якщо використовувати матрицю розміром (2Х2), що складається з координат *X* і *Y* вершин трикутника, то можна записати

$$\begin{bmatrix} 3 & -1 \\ 4 & 1 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 3 \\ -1 & 4 \\ -1 & 2 \end{bmatrix},$$

що є координатами результуючого трикутника $A^{*}B^{*}C^{*}$. Поворот на 180° відносно початку координат досягається шляхом наступного перетворення

$$[\grave{O}] = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix},$$

а на 270° відносно початку координат – перетворенням

$$[\grave{O}] = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}.$$

Зрозуміло матриця тотожного перетворення

$$[\grave{O}] = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

відповідає повороту навколо початку координат на 0° або 360°. Звернемо увагу, що в цих прикладах не зустрічаються масштабування, ні відображення. У цих прикладах здійснюється перетворення в спеціальних випадках повороту навколо початку координат на кути 0°, 90°, 180°, 270°. Як здійснити поворот навколо точки початку координат на довільний кут θ? Для відповіді на це питання розглянемо вектор положення від початку координат до точки *P* (рис.1.2). Позначимо r – довжину вектора, а ϕ – кут між вектором і віссю х. вектор положення повертається навколо початку координат на кут θ та попадає в точку P^*. Записавши вектори положення для *P* и P^*, отримаємо:

$$P = \begin{bmatrix} x & y \end{bmatrix} = \begin{bmatrix} r\cos\phi & r\sin\phi \end{bmatrix}$$

та

$$P^* = \begin{bmatrix} x^* & y^* \end{bmatrix} = \begin{bmatrix} r\cos(\phi+\theta) & r\sin(\phi+\theta) \end{bmatrix}.$$

Використовуючи формулу для косинуса суми кутів, запишемо вираз для Р * такий спосіб.

$$P^* = \begin{bmatrix} x^* & y^* \end{bmatrix} = \begin{bmatrix} r\left(\cos\phi\cos\theta - \sin\phi\sin\theta\right) & r\left(\cos\phi\sin\theta + \sin\phi\cos\theta\right) \end{bmatrix}.$$

Використовуючи визначення для *x* та *y*, можна переписати P^* як

$$P^* = \begin{bmatrix} x^* & y^* \end{bmatrix} = \begin{bmatrix} x\cos\theta - y\sin\theta & x\sin\theta + y\cos\theta \end{bmatrix}.$$

Таким чином, перетворена точка має координати

$$x^* = x\cos\theta - y\sin\theta,$$

$$y^* = x\sin\theta + y\cos\theta$$

Або в матричному вигляді

$$\left[X^*\right] = \left[X\right]\left[T\right] = \left[x^* \quad y^*\right] = \left[x \quad y\right]\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}. \qquad (1.1)$$

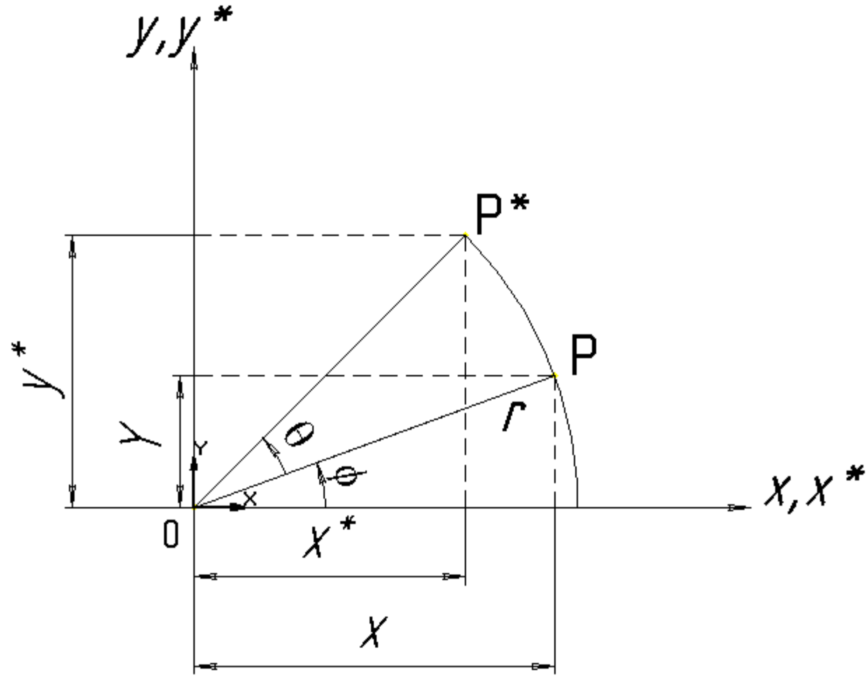

Рис.1.2

Отже, перетворення повороту навколо точки початку координат на довільний кут θ задається матрицею

$$\left[T\right] = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}. \qquad (1.2)$$

Повороти є позитивними, якщо здійснюються проти годинникової стрілки відносно точки обертання (рис. 1.2).

Визначник загальної матриці повороту має наступний вигляд:

$$\det\left[T\right] = \cos^2\theta + \sin^2\theta = 1 \ . \qquad (1.3)$$

У загальному випадку перетворення по матриці з детермінантом, рівним 1, призводять до повного повороту.

Припустимо тепер, що потрібно повернути точку P^* назад в P, тобто виконати зворотне перетворення. Очевидно, що необхідний кут повороту дорівнює – θ. З формули (1.2) візьмемо матрицю для виконання необхідного перетворення

$$[T]^{-1}=\begin{bmatrix}\cos(-\theta) & \sin(-\theta)\\ -\sin(-\theta) & \cos(-\theta)\end{bmatrix}=\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}, \tag{1.4}$$

Так як cos(–θ) = cosθ и sin(–θ) = –sinθ. Вираз $[T]^{-1}$ є формальної записом зворотної матриці $[T]$. Можна показати, що матриця $[T]^{-1}$ є оберненою к $[T]$, якщо згадати, що результат множення матриці на зворотну дає одиничну матрицю. У даному випадку:

$$[T][T]^{-1}=\begin{bmatrix}\cos\theta & \sin\theta\\ -\sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}=$$
$$=\begin{bmatrix}\cos^2\theta+\sin^2\theta & -\cos\theta\sin\theta+\cos\theta\sin\theta\\ -\cos\theta\sin\theta+\cos\theta\sin\theta & \cos^2\theta+\sin^2\theta\end{bmatrix}=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}=[I].$$

де $[I]$ – одинична матрица.

Аналіз виразів (1.2) та (1.4) приводить до іншого цікавого і корисного результату. Згадаймо, що транспонування матриці визначається заміною її рядків стовпцями. Позначимо транспоновану матрицю $[T]$ як $[T]^T$. Порівнюючи її з $[T]^{-1}$, бачимо, що

$$[T]^T=\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}=[T]^{-1}\ . \tag{1.5}$$

Зворотній матриця обертання є транспонованою. Оскільки формально визначник зворотної матриці обчислюється набагато складніше, ніж визначник транспонованою, то вираз (1.5) є досить важливим і корисним результатом. У загальному випадку зворотної для будь матриці перетворення повного повороту, тобто матриці з визначником, рівним +1, є її транспонована матриця (такі матриці називають ортогональними).

1.2 Відображення

У той час, як повний поворот на площині xy зазвичай здійснюється в двовимірному просторі щодо нормалі до площини, відображення являє собою той же поворот на кут 180^o в тривимірному просторі і назад на площину щодо осі, що лежить на площині xy. На рис.1.3 наведені приклади двох відображень на площині трикутника DEF. Відображення відносно прямої $y=0$ (вісь x) отримано з використанням матриці

$$[T]=\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}. \qquad (1.6)$$

У цьому випадку нові вершини трикутника D*E*F* визначатимуться перетворенням

$$\begin{bmatrix}8 & 1\\ 7 & 3\\ 6 & 2\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}=\begin{bmatrix}8 & -1\\ 7 & -3\\ 6 & -2\end{bmatrix}.$$

Подібним чином відображення відносно осі y при $x=0$ буде мати вигляд

$$[T]=\begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix}. \qquad (1.7)$$

Відображення відносно прямої $y=x$ здійснюється за допомогою матриці

$$[T]=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}. \qquad (1.9)$$

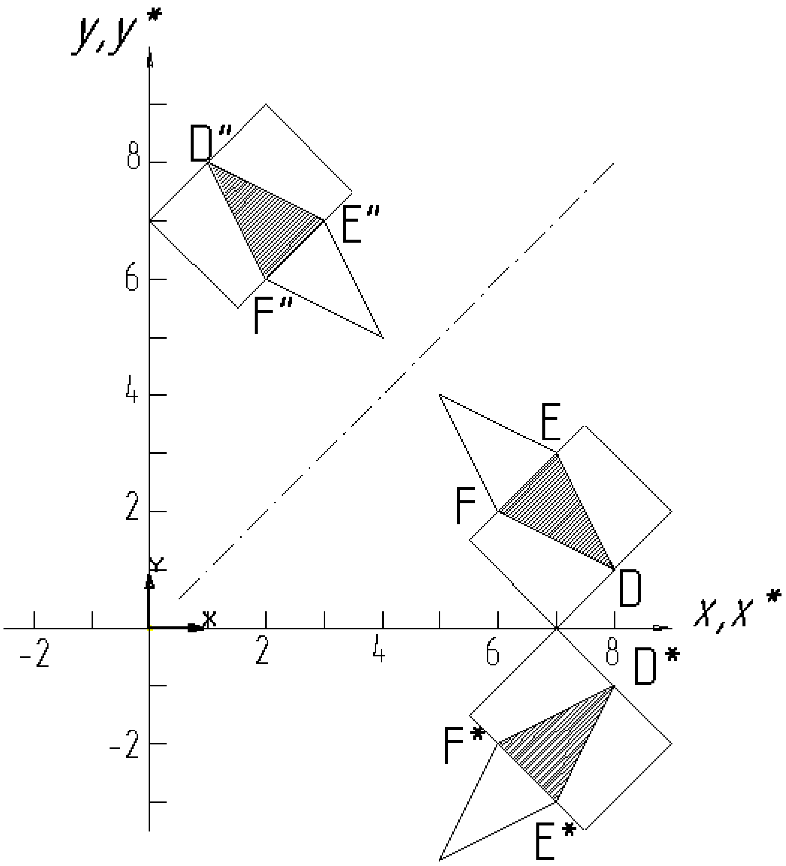

Рис.1.3

Виконавши перетворення, одержимо координати вершин трикутника D*E*F*

$$\begin{bmatrix} 8 & 1 \\ 7 & 3 \\ 6 & 2 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 8 \\ 3 & 7 \\ 2 & 6 \end{bmatrix}.$$

Аналогічним чином відображення відносно осі x буде мати вигляд

$$[T] = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}. \tag{1.10}$$

У кожної з цих матриць визначник дорівнює -1. У загальному випадку, якщо визначник дорівнює -1, то перетворення дає повне відображення.

Якщо обидва повних відображення здійснюються послідовно щодо прямих, що проходять через початок координат, то результатом буде повний поворот відносно початку координат. Це можна побачити, звернувшись до наступного прикладу.

1.3 Масштбування

З міркувань щодо перетворення точок випливає, що величина масштабування визначається значенням елементів вихідної діагональної матриці. Якщо матриця

$$[T]=\begin{bmatrix}2 & 0\\ 0 & 2\end{bmatrix}$$

використовується в якості оператора впливу на вершини трикутника, то має місце «двоupразове» розширення чи рівномірний масштабування щодо точки початку координат. Якщо значення елементів не рівні, то трикутник спотворюється, що зображено на рис.1.4.

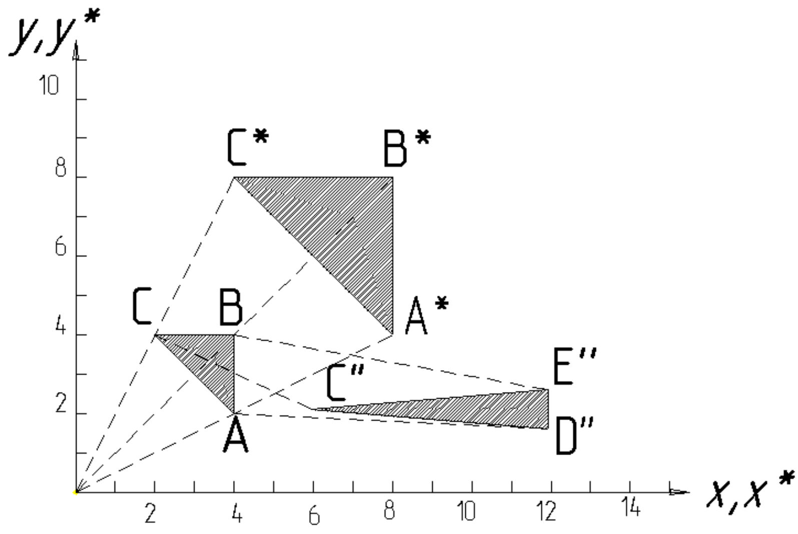

Рис.1.4

Трикутник *ABC*, перетворений за допомогою матриці

$$[T]=\begin{bmatrix}2 & 0\\ 0 & 2\end{bmatrix},$$

переходить в пропорційно збільшений трикутник A*B*C*. Той самий трикутник, але перетворений за допомогою матриці

$$\begin{bmatrix}1/2 & 0\\ 0 & 3\end{bmatrix}$$

переходить в трикутник D*E*F*, що має спотворення, викликане різними коефіцієнтами масштабування. У загальному випадку при матриці

$$[T]=\begin{bmatrix} a & b \\ c & d \end{bmatrix}, \tag{1.11}$$

в якій $a=d,\ b=c=0$, виконується пропорційне масштабування; якщо $a\neq d$, $b=c=0$, то масштабування буде проведено непропорційно. В першому випадку $a=d>1$ відбувається розширення, тобто збільшення зображення. Якщо $a=d<1$, то відбувається рівномірне стиснення, тобто фігура зменшується. Непропорційне розширення і стиснення виникають в залежності від значень a та d, які можуть бути менше або більше, ніж 1, незалежно один від одного. З рис. 1.4 видно також, що на перший погляд перетворення трикутника є переміщенням. Це пояснюється тим, що відносно початку координат масштабуються координатні вектори, а не точки. Для того щоб краще зрозуміти цей факт, розглянемо перетворення ABC в $D^*E^*F^*$ більш уважно. Зокрема,

$$\left[X^*\right]=[X][T]=\begin{bmatrix} 4 & 2 \\ 4 & 4 \\ 2 & 4 \end{bmatrix}\begin{bmatrix} 1/2 & 0 \\ 0 & 3 \end{bmatrix}=\begin{bmatrix} 2 & 6 \\ 2 & 12 \\ 1 & 12 \end{bmatrix}.$$

Зауважимо, що кожна з компонент x координатних векторів трикутника DEF множилася на масштабний коефіцієнт 3, а компоненти y - на 2. Для того щоб отримати чисте масштабування без ефекту переміщення, центр фігури треба помістити в початок координат. Це видно на рис.1.5, на якому трикутник ABC збільшується в два рази при масштабування щодо його центру з координатами, рівними 1/3 підстави і 1/3 висоти. Конкретна матриця перетворення має вигляд

$$\left[X^*\right]=[X][T]=\begin{bmatrix} -1 & -1 \\ 2 & -1 \\ -1 & 2 \end{bmatrix}\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}=\begin{bmatrix} -2 & -2 \\ 4 & -2 \\ -2 & 4 \end{bmatrix}.$$

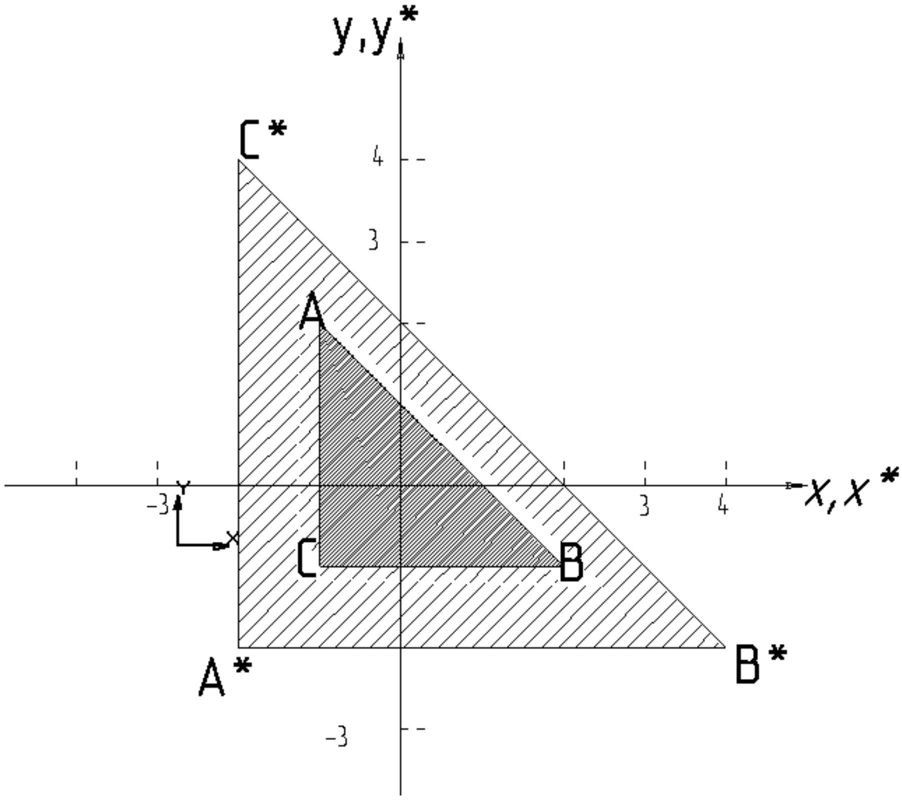

Рис.1.5

2 ДОСЛІДЖЕННЯ ДВОВИМІРНИХ ГЕОМЕТРИЧНИХ ПЕРЕТВОРЕНЬ

Вихідні дані:

```
M=250; N=400; H=80; L=80; Alfa=0;
Tetaz=0; Tetay=10; Tetax=30;
Tz=0; Ty=40; Tx=25;
Sz=1.5; Sy=1.5; Sx=1.2;
```

2.1. Сформуємо двомірне зображення заданої геометричної фігури

```
% формированаие исходного изображения
Im2D=zeros(M,N);
Im2D(round(M/2-L/2):round(M/2+L/2),round(N/2-H/2):round(N/2+H/2))=1;
Im2ix = Im2D;
Im2D=imrotate(Im2D,Alfa,'crop'); % в градусах против час стр
figure; imshow(Im2D); title('ISX IZOB');
```

Результат представлений на рис.2.1

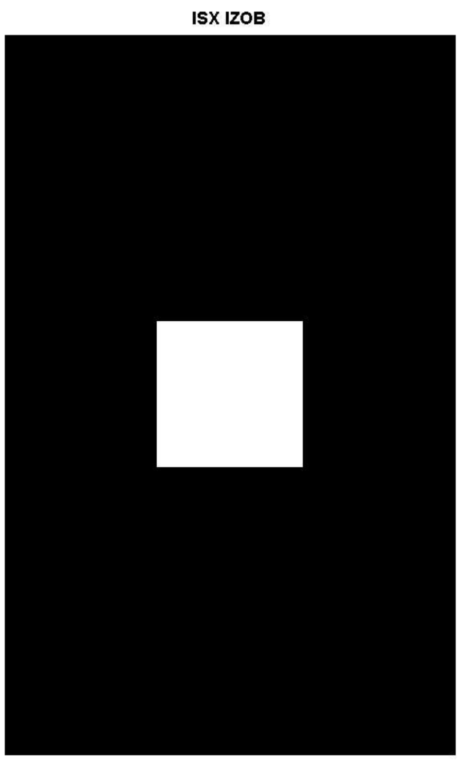

Рис.2.1

2.2 Виділимо 3 контрольні точки геометричної Фігури і дізнаємося їх координати

а) шляхом обчислень

Контрольними точками виберемо кути прямокутника і розрахуємо їх координати до повороту фігури на 60 градусів, результат представлений на рис.2.2

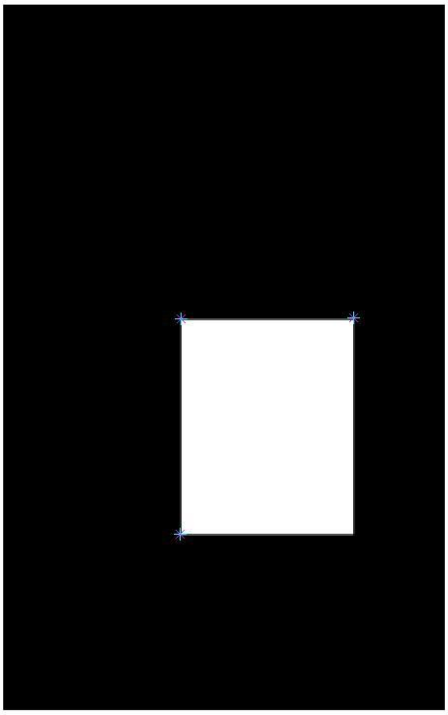

Рис.2.2

Лівий верхній кут **(85, 160);**

X = 2500/2-80/2 = 85

Y = 400/2 - 80/2 =160

Лівий нижній кут **(85, 240);**

X =250/2-80/2 = 85

Y = 400/2+ 80/2 = 240

Правий верхній кут (165, 160);

X = 250/2 +80/2 = 165

Y = 400/2- 80/2= 160

б) шляхом вимірювань за допомогою функції **impixel(Im2D)**, яка представляє графічний інструмент для вибору контрольних точок під назвою Control Point Selection Tool

```
XIsx =

    84
    84
   166


YIsx =

   159
   241
   159
```

Координати цих точок після повороту фігури на 60 градусів проти годинникової стрілки

Лівий верхній кут

X = (85-125) cos(-60) - (160-200) sin (-60) + 150= 70

Y = (85-125) sin(-60) + (160-200) cos (-60) + 200 =210

Лівий нижній кут

X = (85-150) cos(-60) - (240-200) sin (-60) + 150 = 140

Y =(85-150) sin(-60) + (240-200) cos (-60) + 200 =255

Правий верхній кут

X = (165-125) cos(-60) - (160-200) sin (-60) + 250 =110

Y = (165-125) sin(-60) + (160-200) cos (-60) + 200 =145 б) шляхом вимірювань за допомогою функції **impixel(Im2D)**, яка представляє графічний інструмент для вибору контрольних точок під назвою Control Point Selection Tool , (рис.2.3)

impixel(Im2D)

```
XIsx =


    71
   141
   111

YIsx =


   214
   250
   147
```

Шляхом розрахунку	**impixel(Im2D)**
Лівий верхній кут (70, 210); Лівий нижній кут (140, 255); Правий верхній кут (110, 145);	Лівий верхній кут (71, 214); Лівий нижній кут (141, 250); Правий верхній кут (111, 147);

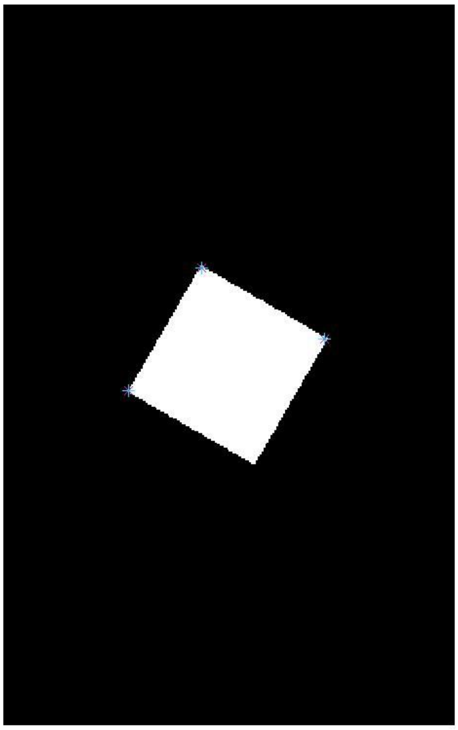

Рис.2.3

Порівнюючи результати, отримані шляхом розрахунку і шляхом вибору за допомогою **impixel(Im2D)**, ми бачимо незначне відмінність у координатах. Цей факт обумовлений похибкою яку вносить людина, яка обирає контрольні точки за допомогою графічного інструменту, а також пристроєм введення і відображення графічної інформації.

2.3 Маштабування та переміщення прямокутника

Маштабування

$$\begin{cases} X = k_x \cdot x_0 \\ Y = k_y \cdot y_0 \end{cases}$$

Переміщення

$$\begin{cases} X = x + dx \\ Y = y + dy \end{cases}$$

$k_x = 1{,}2;\ k_y = 1{,}5 \quad dx = 30;\ dy = 10$

Лівий верхній кут

X = 1,2 · 70= 84

Y = 1,5 · 210 = 315

Лівий верхній кут

X = 70 + 30 = 100

Y = 210 + 10 = 220

Лівий нижній кут

X = 1,2 · 140 = 168

Y = 1,5 · 255 = 383

Лівий нижній кут

X = 140 + 30 = 170

Y = 255 + 10 = 265

Правий верхній кут

X = 1,2 · 110 = 132

Y = 1,5 · 145 = 218

Правий верхній кут

X = 110 + 30 = 120

Y = 145 + 10 =155

2.4 Визначимо матрицю та операцію геометричного перетворення в MATLAB та отримаємо зображення-результат перетворення рис.2.4

```
% матриці прямого перетворення
R2D=[cos(Tetaz*pi/180) sin(Tetaz*pi/180) 0
    -sin(Tetaz*pi/180) cos(Tetaz*pi/180) 0
     0                 0                 1]; % поворот за часовою стрілкою
S2D=[Sx  0   0
     0   Sy  0
     0   0   1]; % розтяг
T2D=[1   0   0
     0   1   0
     Tx  Ty  1]; % переміщення
TFormR2D=maketform('affine',R2D);
TFormS2D=maketform('affine',S2D);
TFormT2D=maketform('affine',T2D);

% пряме геометричне перетворення вхідного зображення
Im2DTransform1=imtransform(Im2D,TFormR2D,...
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);
Im2DTransform2=imtransform(Im2DTransform1,TFormS2D,...
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);
Im2DTransform3=imtransform(Im2DTransform2,TFormT2D,...
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);
```

```
% вивід результатів перетворення
figure; subplot(2,2,1); imshow(Im2D); title('Вхідне зображення');
subplot(2,2,2); imshow(Im2DTransform1); title('Поворот');
subplot(2,2,3); imshow(Im2DTransform2); title('Маштабування');
subplot(2,2,4); imshow(Im2DTransform3); title('Переміщення');
```

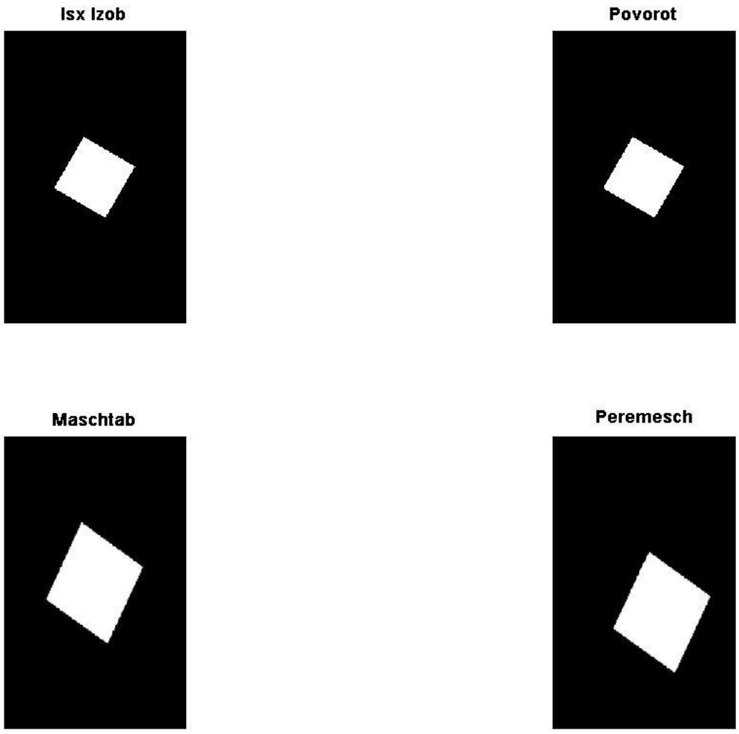

Рис.2.4

Матриця геометричного перетворення відповідно програмі обчислень:

```
PTransform3 =

    0.0327    0.0327    0.0327
         0         0         0
    1.0000    1.0000    1.0000
```

2.5 Визначимо на зображенні-результаті перетворення координат контрольних точок за допомогою *impixel*, та порівняємо їх з даними, отриманими при виконанні пункту 2.3

```
[XTransform3, YTransform3, PTransform3]=impixel(Im2DTransform3)

XTransform3 =

   86
   170
   135

YTransform3 =

  320
  380
  219
```

Шляхом розрахунку	impixel(Im2DTransform3)
Лівий верхній кут (84, 315); Лівий нижній кут (168, 383); Правий верхній кут (132, 218);	Лівий верхній кут (86, 170); Лівий нижній кут (170, 350); Правий верхній кут (135, 219);

Так само, як і при виконанні пункту 2.2, отримані дані мають відмінність. Це зумовлено наявністю інструментальної похибки та похибкою яку вносить людина, яка обирає контрольні точки за допомогою графічного інструменту, а також пристроєм введення і відображення графічної інформації.

2.6 Виконати в MATLAB зворотне геометричне перетворення шляхом:

а) застосування матриці оберненого перетворення відносно результатів п.2.4. представлено на рис.2.5.

Матриця оберненого геометричного перетворення:

```
PRestore3 =

    0.1444    0.1444    0.1444
    0.0889    0.0889    0.0889
    1.0000    1.0000    1.0000
```

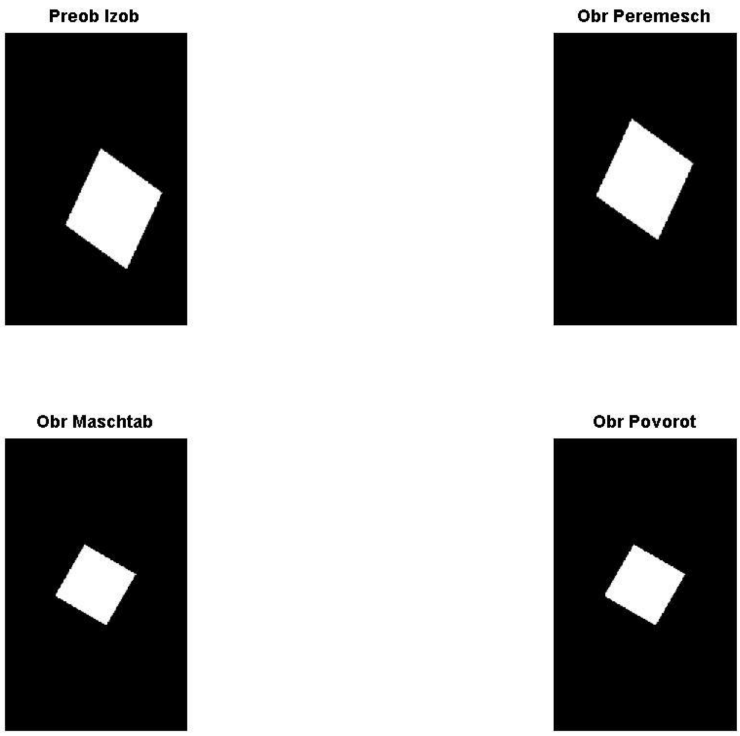

Рис.2.5

б) Визначимо на зображенні, відновленому шляхом зворотнього геометричного перетворення, координати контрольних точок за допомогою *impixel(Im2DRestore3)*, та порівняємо їх з результатами п. 2.2.

```
[XRestore3, YRestore3, PRestore3]=impixel(Im2DRestore3)

   XRestore3 =

      111
      176
      126


   YRestore3 =

     200
      245
      153
```

Шляхом розрахунку	**impixel(Im2DRestore3)**
Лівий верхній кут (100, 220);	(111, 200);
Лівий нижній кут (170, 265);	(176, 245);
Правий верхній кут (120, 155);	(126, 153);

Так само, як і при виконанні пункту 2.2 і 2.5 , отримані дані мають відмінність. Це зумовлено наявністю інструментальної похибки та похибкою яку вносить людина, яка обирає контрольні точки за допомогою графічного інструменту, а також пристроєм введення і відображення графічної інформації.

3 РОЗРОБКА МЕТОДУ ГЕОМЕТРИЧНИХ ПЕРЕТВОРЕНЬ ЗОБРАЖЕНЬ ТРИВИМІРНИХ ОБЄКТІВ

3.1 Сформуємо літер згідно варіанту завдання таким чином , щоб вони були розташовані в растрі розміром NxM дискретних точок (рис.3.1)

Вихідні дані:

```
M=250; N=400; H=80; L=80; Alfa=0;
Tetaz=0; Tetay=10; Tetax=30;
Tz=0; Ty=40; Tx=25;
Sz=1.5; Sy=1.5; Sx=1.2;

    % формированаие исходного изображения

Im2D=imread('Eii.png');
Im2D=im2double(Im2D(1:M,1:N));
Im2D=imrotate(Im2D,Alfa,'crop'); % в градусах против час стр
figure; imshow(Im2D); title('ISX IZOB');
```

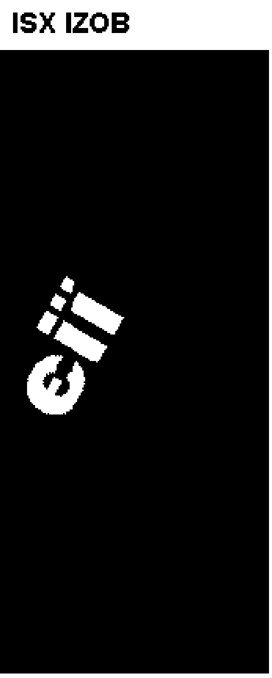

Рис.3.1

Для зображення тривимірного об’єкту досліджуємо геометричне перетворення в тривимірному просторі згідно мого варіанту на основі 4 контрольних точок

а) виконаємо проекції: повороту, масштабування, переміщення (рис.3.2):

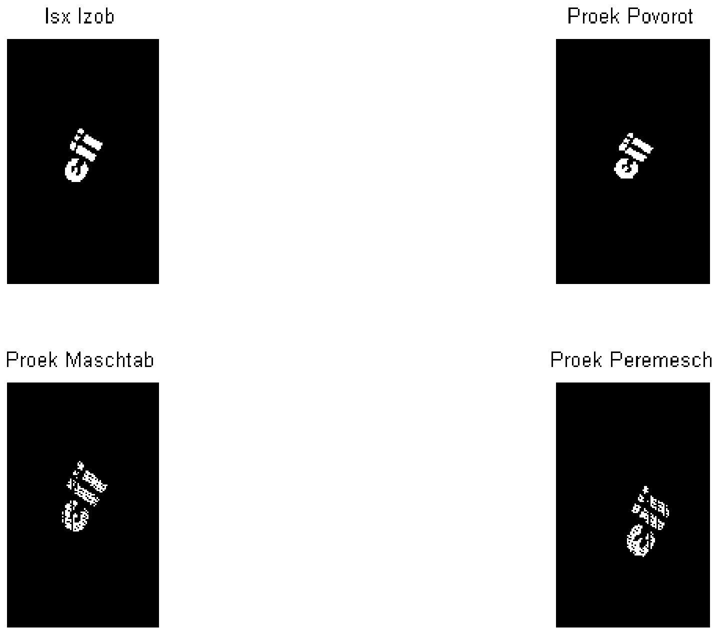

Рис.3.2

б) Виконуємо зворотне геометричне перетворення перетвореного зображення (рис.3.3):

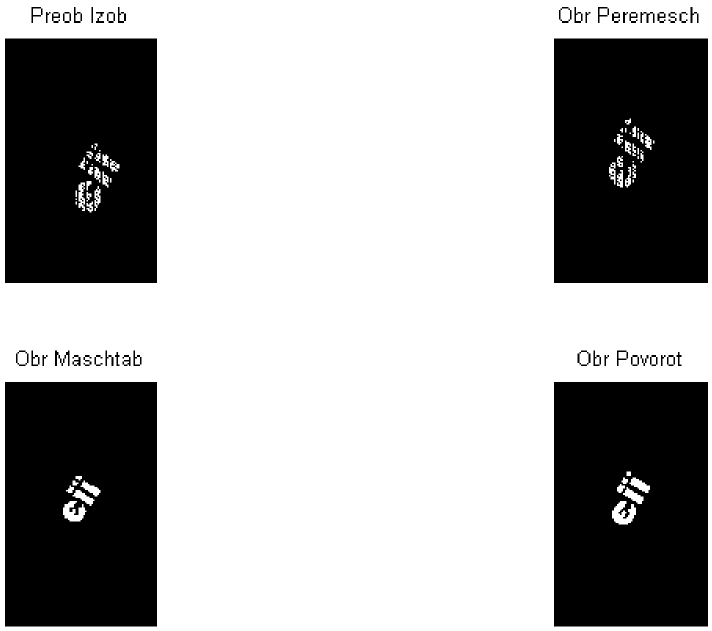

Рис.3.3

в) На основі 4 контрольних точок дослідимо геометричне перетворення в тривимірному просторі за допомогою **impixel**:

```
[XIsx, YIsx, PIsx]=impixel(Im2D)
```

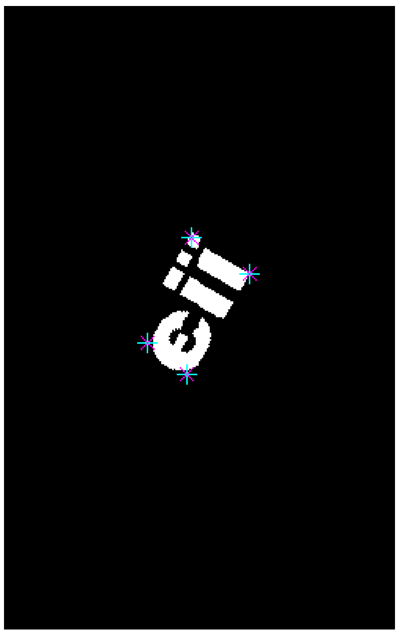

Рис.3.4

```
[XTransform3, YTransform3, PTransform3]=impixel(Im2DTransform3)
```

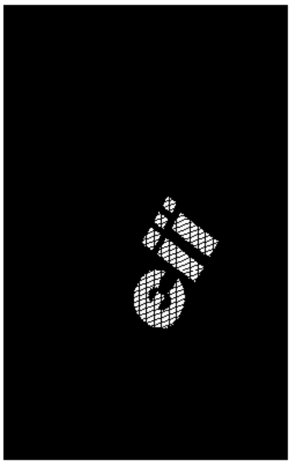

Рис.3.5

```
[XRestore3, YRestore3, PRestore3]=impixel(Im2DRestore1)
```

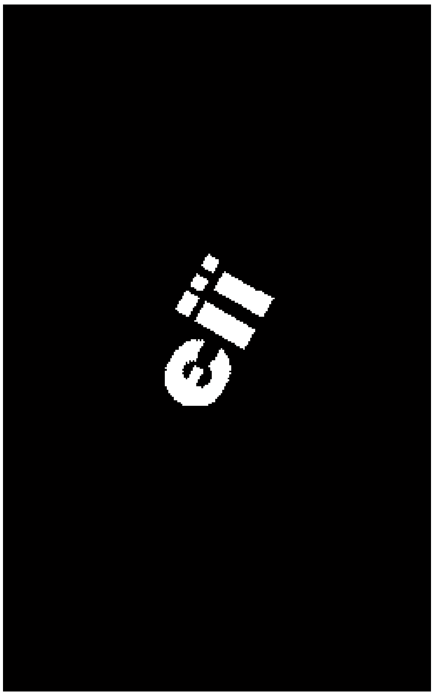

Рис.3.6

```
XIsx =
    92
   117
   120
   158

YIsx =
   217
   237
   149
   172

PIsx =

     0     0     0
     0     0     0
     1     1     1
     1     1     1



XTransform3 =

   114
   145
   144
   190



YTransform3 =

   260
   283
   171



PRestore3 =

     1     1     1
     0     0     0
     0     0     0
     1     1     1
```

```
   211



PTransform3 =
     0     0     0
     0     0     0
     1     1     1
     0     0     0



XRestore3 =
    90
   119
   120
   157



YRestore3 =
   218
   239
   152
   172



PRestore3 =
     0     0     0
     0     0     0
     1     1     1
     1     1     1
```

Порівняння координат початкового та отриманого зображення після перетворень:

XIsx = 92 117 120 158 YIsx = 217 237 149 172	XRestore3 = 90 119 120 157 YRestore3 = 218 239 152 172

При дослідженні тривимірних зображень отримані дані відмінні від початкових, що зумовлено наявністю інструментальної похибки та похибкою яку вносить людина, яка обирає контрольні точки за допомогою графічного інструменту, а також пристроєм введення і відображення графічної інформації.

ВИСНОВКИ

В даній курсовій роботі були досліджені методи геометричних перетворень об’єктів. В теоретичній частині розглядалися види геометричних перетворень такі, як переміщення, масштабування, обертання. В практичній частині за допомогою цих видів досліджували геометричні перетворення двовимірних та тривимірних зображень. В процесі досліджування координати перетворення розраховували згідно існуючих формул та за допомогою існуючої програми. В результаті отримали деяку відмінність значень, що була зумовлена наявністю інструментальної похибки та похибкою яку вносить людина, яка обирає контрольні точки за допомогою графічного інструменту, а також пристроєм введення і відображення графічної інформації.

ДОДАТКИ

Програма для двовимірного перетворення

```
% исходные данные
clear all;
M=250; N=400; H=80; L=80; Alfa=0;
Tetaz=0; Tetay=10; Tetax=30;
Tz=0; Ty=40; Tx=25;
Sz=1.5; Sy=1.5; Sx=1.2;

% формированаие исходного изображения
Im2D=zeros(M,N);
Im2D(round(M/2-L/2):round(M/2+L/2),round(N/2-H/2):round(N/2+H/2))=1;
Im2D=imrotate(Im2D,Alfa,'crop'); % в градусах против час стр
figure; imshow(Im2D); title('ISX IZOB');

% матрицы прямого геометрического преобразования
R2D=[cos(Tetaz*pi/180) sin(Tetaz*pi/180) 0
    -sin(Tetaz*pi/180) cos(Tetaz*pi/180) 0
     0                 0                 1]; % в рад по час стр
S2D=[Sx  0   0
     0   Sy  0
     0   0   1]; % растяжение раз
T2D=[1   0   0
     0   1   0
     Tx  Ty  1]; % перенос вправо и вниз
TFormR2D=maketform('affine',R2D);
TFormS2D=maketform('affine',S2D);
TFormT2D=maketform('affine',T2D);

% прямое геометрическое преобразование исх изображения
Im2DTransform1=imtransform(Im2D,TFormR2D,...
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);
Im2DTransform2=imtransform(Im2DTransform1,TFormS2D,...
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);
Im2DTransform3=imtransform(Im2DTransform2,TFormT2D,...
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);

% вывод преобразованного изображения
figure; subplot(2,2,1); imshow(Im2D); title('Isx Izob');
subplot(2,2,2); imshow(Im2DTransform1); title('Povorot');
subplot(2,2,3); imshow(Im2DTransform2); title('Maschtab');
subplot(2,2,4); imshow(Im2DTransform3); title('Peremesch');

% матрицы обратного геометрического преобразования
R2DInv=[cos(Tetaz*pi/180) -sin(Tetaz*pi/180) 0
        sin(Tetaz*pi/180)  cos(Tetaz*pi/180) 0
        0                  0                 1]; % в рад по час стр
S2DInv=[1/Sx 0    0
        0    1/Sy 0
        0    0    1]; % растяжение раз
T2DInv=[1    0    0
        0    1    0
        -Tx  -Ty  1]; % перенос вправо и вниз
TFormT2DInv=maketform('affine',T2DInv);
TFormS2DInv=maketform('affine',S2DInv);
TFormR2DInv=maketform('affine',R2DInv);

% обратное геометрическое преобразование преоб изображения
Im2DRestore1=imtransform(Im2DTransform3,TFormT2DInv,...
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);
Im2DRestore2=imtransform(Im2DRestore1,TFormS2DInv,...
```

```
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);
Im2DRestore3=imtransform(Im2DRestore2,TFormR2DInv,...
    'UData',[-round(N/2)+1 round(N/2)],'VData',[-round(M/2)+1 round(M/2)],...
    'XData',[-round(N/2)+1 round(N/2)],'YData',[-round(M/2)+1 round(M/2)]);

% вывод восстановленного изображения
figure; subplot(2,2,1); imshow(Im2DTransform3); title('Preob Izob');
subplot(2,2,2); imshow(Im2DRestore1); title('Obr Peremesch');
subplot(2,2,3); imshow(Im2DRestore2); title('Obr Maschtab');
subplot(2,2,4); imshow(Im2DRestore3); title('Obr Povorot');

% получение координат точек
figure; [XIsx, YIsx, PIsx]=impixel(Im2D)
[XTransform3, YTransform3, PTransform3]=impixel(Im2DTransform3)
[XRestore3, YRestore3, PRestore3]=impixel(Im2DRestore3)
```

Програма для тривимірного перетворення

```
% исходные данные
clear all;
M=250; N=400; H=80; L=80; Alfa=0;
Tetaz=0; Tetay=10; Tetax=30;
Tz=0; Ty=40; Tx=25;
Sz=1.5; Sy=1.5; Sx=1.2;

% формированаие исходного изображения
Im2D=imread('Eii.png');
Im2D=im2double(Im2D(1:M,1:N));
%Im2D=zeros(M,N);
%Im2D(round(M/2-L/2):round(M/2+L/2),round(N/2-H/2):round(N/2+H/2))=1;
Im2D=imrotate(Im2D,Alfa,'crop'); % в градусах против час стр
figure; imshow(Im2D); title('ISX IZOB');

% матрицы прямого геометрического преобразования
R3DX=[1  0                  0
    0  cos(Tetax*pi/180) sin(Tetax*pi/180) 0
      0 -sin(Tetax*pi/180) cos(Tetax*pi/180) 0
      0  0                  0                 1]; % в рад по час стр
R3DY=[cos(Tetay*pi/180)    0                 -sin(Tetay*pi/180) 0
     0                    1                 0                  0
      sin(Tetay*pi/180)    0                 cos(Tetay*pi/180) 0
     0                    0                 0                  1]; % в рад по час стр
R3DZ=[cos(Tetaz*pi/180) sin(Tetaz*pi/180) 0  0
     -sin(Tetaz*pi/180) cos(Tetaz*pi/180) 0  0
      0                  0                 1  0
     0                  0                 0  1]; % в рад по час стр
R3D=R3DZ*R3DY*R3DX;
S3D=[Sx  0   0   0
     0   Sy  0   0
     0   0   Sz  0
    0   0   0   1]; % растяжение раз
T3D=[1   0   0   0
     0   1   0   0
     0   0   1   0
     Tx  Ty  Tz  1]; % перенос вправо и вниз


TFormR3D=maketform('affine',R3D);
TFormS3D=maketform('affine',S3D);
TFormT3D=maketform('affine',T3D);
MP3D=[1   0   0   0
     0   1   0   0
     0   0   0   0
     0   0   0   1]; % ортографическая проекция

TFormMP3D=maketform('projective',MP3D);

% прямое геометрическое преобразование исх изображения
```

```
A3D=zeros(M*N,3);
k=0;
for j=1:M
    for i=1:N
        if Im2D(j,i)>=0.5
            k=k+1;
            A3D(k,1)=i-round(N/2)+1; A3D(k,2)=j-round(M/2)+1;
        end;
    end;
end;
B3D=A3D(1:k,:);
C3D1=tformfwd(TFormR3D, B3D);
C3D2=tformfwd(TFormS3D, C3D1);
C3D3=tformfwd(TFormT3D, C3D2);
D3D1=tformfwd(TFormMP3D, C3D1);
D3D2=tformfwd(TFormMP3D, C3D2);
D3D3=tformfwd(TFormMP3D, C3D3);
Im2DTransform1=zeros(M,N);
for i=1:k
    x=round(D3D1(i,1)+round(N/2)-1); y=round(D3D1(i,2)+round(M/2)-1);
    Im2DTransform1(y,x)=1;
end;
Im2DTransform2=zeros(M,N);
for i=1:k
    x=round(D3D2(i,1)+round(N/2)-1); y=round(D3D2(i,2)+round(M/2)-1);
    Im2DTransform2(y,x)=1;
end;
Im2DTransform3=zeros(M,N);
for i=1:k
    x=round(D3D3(i,1)+round(N/2)-1); y=round(D3D3(i,2)+round(M/2)-1);
    Im2DTransform3(y,x)=1;
end;
% вывод преобразованного изображения
figure; subplot(2,2,1); imshow(Im2D); title('Isx Izob');
subplot(2,2,2); imshow(Im2DTransform1); title('Proek Povorot');
subplot(2,2,3); imshow(Im2DTransform2); title('Proek Maschtab');
subplot(2,2,4); imshow(Im2DTransform3); title('Proek Peremesch');

% % матрицы обратного геометрического преобразования
 R3DXInv=[1  0                 0                 0
          0  cos(Tetax*pi/180) -sin(Tetax*pi/180) 0
          0  sin(Tetax*pi/180)  cos(Tetax*pi/180) 0
          0  0                 0                 1]; % в рад по час стр
 R3DYInv=[cos(Tetay*pi/180)     0                 sin(Tetay*pi/180)  0
          0                     1                 0                  0
         -sin(Tetay*pi/180)     0                 cos(Tetay*pi/180)  0
          0                     0                 0                  1]; % в рад по час
стр
 R3DZInv=[cos(Tetaz*pi/180) -sin(Tetaz*pi/180) 0  0
          sin(Tetaz*pi/180)  cos(Tetaz*pi/180) 0  0
          0                  0                 1  0
          0                  0                 0  1]; % в рад по час стр
 R3DInv=R3DZInv*R3DYInv*R3DXInv;

 S3DInv=[1/Sx  0     0     0
         0     1/Sy  0     0
         0     0     1/Sz  0
         0     0     0     1]; % растяжение раз

 T3DInv=[1    0    0    0
         0    1    0    0
         0    0    1    0
        -Tx -Ty -Tz  1]; % перенос вправо и вниз
 TFormR3DInv=maketform('affine',R3DInv);
 TFormS3DInv=maketform('affine',S3DInv);
 TFormT3DInv=maketform('affine',T3DInv);

% обратное геометрическое преобразование преоб изображения
C3D2Restore=tforminv(TFormT3D, C3D3);
C3D1Restore=tforminv(TFormS3D, C3D2Restore);
B3DRestore=tforminv(TFormR3D, C3D1Restore);
D3D3Restore=tformfwd(TFormMP3D, C3D2Restore);
```

```
D3D2Restore=tformfwd(TFormMP3D, C3D1Restore);
D3D1Restore=tformfwd(TFormMP3D, B3DRestore);
Im2DRestore3=zeros(M,N);
for i=1:k
    x=round(D3D3Restore(i,1)+round(N/2)-1); y=round(D3D3Restore(i,2)+round(M/2)-1);
    Im2DRestore3(y,x)=1;
end;
Im2DRestore2=zeros(M,N);
for i=1:k
    x=round(D3D2Restore(i,1)+round(N/2)-1); y=round(D3D2Restore(i,2)+round(M/2)-1);
    Im2DRestore2(y,x)=1;
end;
Im2DRestore1=zeros(M,N);
for i=1:k
    x=round(D3D1Restore(i,1)+round(N/2)-1); y=round(D3D1Restore(i,2)+round(M/2)-1);
    Im2DRestore1(y,x)=1;
end;

% вывод восстановленного изображения
figure; subplot(2,2,1); imshow(Im2DTransform3); title('Preob Izob');
subplot(2,2,2); imshow(Im2DRestore3); title('Obr Peremesch');
subplot(2,2,3); imshow(Im2DRestore2); title('Obr Maschtab');
subplot(2,2,4); imshow(Im2DRestore1); title('Obr Povorot');

% получение координат точек
figure; [XIsx, YIsx, PIsx]=impixel(Im2D)
[XTransform3, YTransform3, PTransform3]=impixel(Im2DTransform3)
[XRestore3, YRestore3, PRestore3]=impixel(Im2DRestore1)
```