Universidad Nacional de San Agustín
Ciencia De La Computación

TRANSFORMACIONES INDUCIDAS POR MATRICES

Matematica Aplicada a la Computación

Quincho Mamani, Lehi
Cui: 20122586

Oct - 2017

RESUMEN

En el presente trabajo se implementará en un lenguaje de programación las transformaciones inducidas por matrices, para luego mostrar los el proceso de estas operaciones en forma grafica por medio una librería grafica, y algunas herramientas de programación.

HERRAMIENTAS

Lenguaje de Programación: C++
Open Graphics Library (OpenGL)
Qt Framework

DEFINICIÓN

Dada una matriz A *de* $m \times n$, *La transformación inducida por* A *es* :

$T_A : \mathbb{R}^n \rightarrow \mathbb{R}^m$

$x_{nx1} \Rightarrow A_{mxn}\, x_{nx1}$

$T_A(x) = Ax$

EXPLICACIÓN DEL CODIGO

Usando la librería grafica OpenGL, y el framework QT , tenemos estas siguientes funciones que nos ayudarán a visualizar de forma grafica las operaciones.

```
void initializeGL();    //Inicializa los paramtros y variables del ViewPort
                        (interfaz) de OpenGl
void paintGL();         //Esta función es llamada constantemente para graficar
                        los cambios realizados a tiempo real. Esta es la función
                        que se encargará de dibujar las rectas y puntos.
void resizeGL(int w, int h);  //Esta función sirve para acomodar las dimensiones
                              de la pantalla, y algunos otros parametros más.
```

Esta función será llamada cuando cuando se haga el evento del Click en el boton de agregar nuevo Punto (Vector). Aqui se reciven los parametros necesarios tales como la Matriz de transformación, los vectores a transformar, la dimension original, y la dimensión a la que se va a transformar.

```
void MainWindow::on_newPointButton_clicked()
{

    try
    {

        string::size_type sz;

//En las siguientes lineas se capturan los valores ingresados desde la interfaz
de usuario , dim es la dimensión original, DimRes es la dimensión a la que se va
a transformar.
```

```
    int dim=atoi(ui->dim1->currentText().toStdString().c_str());
    int dimRes=atoi(ui->dim2->currentText().toStdString().c_str());

    float x=stof(ui->p_x->text().toStdString().c_str(),&sz);
    float y=stof(ui->p_y->text().toStdString().c_str(),&sz);

    OVector *vec_org = new Ovector(dim);

    // Se verifica la dimensión para generar un grafico en 2D, o 3D.

    if(dim==3)
    {
        float z=stof(ui->p_z->text().toStdString().c_str(),&sz);

        vec_org->setCoord3D(x,y,z);
        ui->OrWidget->listVec->push_back(vec_org);
        ui->OrWidget->dim=3;
    }
    else
    {
        vec_org->setCoord2D(x,y);
        ui->OrWidget->listVec->push_back(vec_org);
        ui->OrWidget->dim=2;
    }

    float **matrizDeT= new float*[dimRes];

    for(int i=0;i<dimRes;i++)
    {
        matrizDeT[i]=new float[dim];
        for(int j=0;j<dim;j++)
        {
            matrizDeT[i][j]=ui->matrizT->item(i,j)->text().toFloat();
        }
    }

    OVector *vec_res_T=new OVector(dimRes);

        //Una vez recibidos los datos que necesitamos para realizar la
        transformación, llamamos a la función que realizará estas
        operaciones

    transformVectwithMatrix ( dim , dimRes,matrizDeT, vec_org , vec_res_T );

        //Se modifican los datos que serán capturados por la función paintGL
        para ser mostrados en la interfaz.

    ui->FinWidget->listVec->push_back(vec_res_T);
    ui->FinWidget->dim=dimRes;

    //-----------------------------------------------------------------

}
catch(…) //Si ocurre algun error, tales como, no se han llenado los campos
             correctamente, simplemente imprimirá en consola (no en la
             interfaz) un mensaje de erro, esto no se verá desde la
             interfaz, y solo es sirve para fines de prueba.
{
    cout<<"Error"<<endl;
```

```
    }
```

Definimos la función que realizará las operaciones que necesitamos, esta función se llama ***transformVectwithMatrix ,*** y como parametros de entrada tenemos:

```
void transformVectwithMatrix(                    // Función        T_A(x)
                             int dim_ini,        // Dimensión de   IR^n (n)
                             int dim_end,        // Dimensión de   IR^m (m)
                             float **Mat_o_Transf,// Matriz         A_mxn
                             OVector *vec_orig , // Vector         x_nx1
                             OVector * result )  // Resultado      Ax = S_mx1
```

Las siguientes lineas son las que hacen la operación de multiplicación de la Matriz de transformación con el vector original, para dar como resultado a el vector transformado.

```
{

    for(int i=0;i<dim_end;i++) // Esta linea nos ayuda a recorrer las filas de
nuestra matriz

   {

        float temp_res=0; // Esta variable sirve para almacenar los resultados
                          de la multiplicación que se acumularán cuando el
                          algoritmo recorra cada elemento en de cada fila en
                          el ciclo que se muestra en las siguientes lineas.

        for(int j=0;j<dim_ini;j++)  //Esta sentencia recorre cada elemento de
                                    una fila i.
            temp_res+= Mat_o_Transf[i][j]*vec_orig→vec[j];

        result->vec[i]=temp_res;
    }
}
```

RESULTADO