CSE462/562 - Augmented Reality

HW2 - Rapor

PART1:

1.1->

```
public static double[,] calculateHomographyMatrix(double[] x, double[] y, dou
{
    double[,] result = new double[3, 3];
    for (int i = 1; i <= MATRIX_ROWS; i++)
    {
        matrix[2 * i - 1 - 1, 0] = -x[i - 1];
        matrix[2 * i - 1 - 1, 1] = -y[i - 1];
        matrix[2 * i - 1 - 1, 2] = -1;
        matrix[2 * i - 1 - 1, 3] = 0;
        matrix[2 * i - 1 - 1, 4] = 0;
        matrix[2 * i - 1 - 1, 5] = 0;
        matrix[2 * i - 1 - 1, 6] = x[i - 1] * u[i - 1];
        matrix[2 * i - 1 - 1, 7] = u[i - 1] * y[i - 1];
        matrix[2 * i - 1 - 1, 8] = u[i - 1];
        matrix[2 * i - 1, 0] = 0;
        matrix[2 * i - 1, 1] = 0;
        matrix[2 * i - 1, 2] = 0;
        matrix[2 * i - 1, 3] = -x[i - 1];
        matrix[2 * i - 1, 4] = -y[i - 1];
        matrix[2 * i - 1, 5] = -1;
        matrix[2 * i - 1, 6] = x[i - 1] * v[i - 1];
        matrix[2 * i - 1, 7] = v[i - 1] * y[i - 1];
        matrix[2 * i - 1, 8] = v[i - 1];
    }
    var svd = matrix.Svd(true);
    var homgraphy = svd.VT;
    var homography_transpose = homgraphy.Transpose();
    int k = 0;
    for (int i = 0; i < 3; i++)
    { for (int j = 0; j < 3; j++)
        {
            result[i, j] = homography_transpose[k,8];
            k++;
        }
```

1.2->

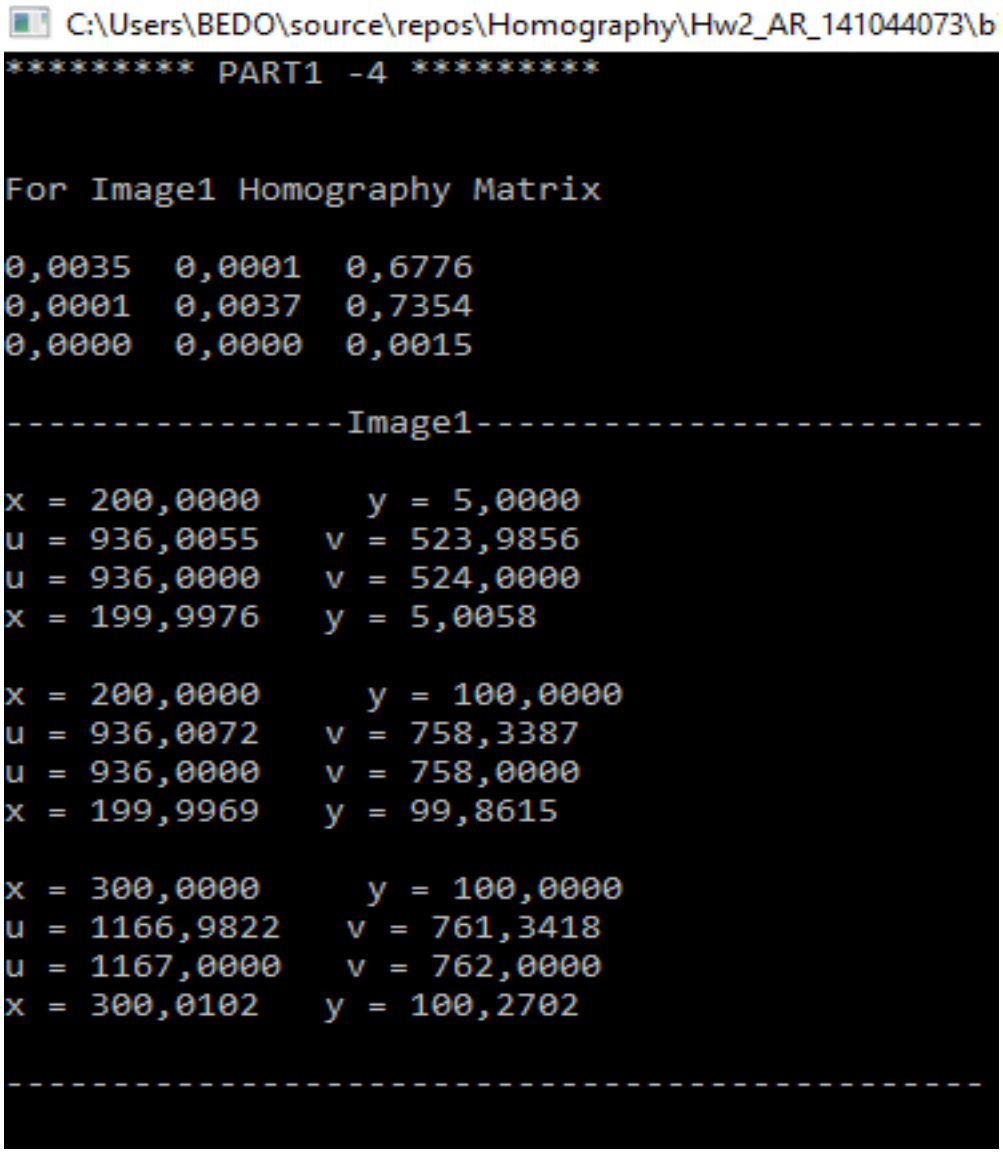
```
public static void calculateTarget(double[,] matrixA,double[,] matrixB)
{
    int aRows = 3;
    int aCols = 3;
    int bRows = 3;
    int bCols = 1;

    double[][] target = MatrixCreate(aRows, bCols);
    for (int i = 0; i < aRows; ++i)
    {
        for (int j = 0; j < bCols; ++j)
        { // each col of B
            for (int k = 0; k < aCols; ++k)
            {
                target[i][j] += matrixA[i,k] * matrixB[k,j];


            }


        }

    }
    target[0][0] = target[0][0] / target[2][0];
    target[1][0] = target[1][0] / target[2][0];

    Console.Write("u = " + target[0][0].ToString("0.0000") + "    ");
    Console.Write("v = " + target[1][0].ToString("0.0000"));
    Console.WriteLine();
    Console.WriteLine();
```

1.3->

```
public static void calculateSource(double[,] matrixA, double[,] matrixB)
{
    var tmp = DenseMatrix.OfArray(matrixA);
    var h_inverse = tmp.Inverse();
    calculateSource_(h_inverse.ToArray(), matrixB);
}
public static void calculateSource_(double[,] matrixA, double[,] matrixB)
{
    int aRows = 3;
    int aCols = 3;
    int bRows = 3;
    int bCols = 1;

    double[][] target = MatrixCreate(aRows, bCols);
    for (int i = 0; i < aRows; ++i)
    {
        for (int j = 0; j < bCols; ++j)
        { // each col of B
            for (int k = 0; k < aCols; ++k)
            {
                target[i][j] += matrixA[i, k] * matrixB[k, j];


            }

        }

    }
    target[0][0] = target[0][0] / target[2][0];
    target[1][0] = target[1][0] / target[2][0];

    Console.Write("x = " + target[0][0].ToString("0.0000") + "    ");
    Console.Write("y = " + target[1][0].ToString("0.0000"));
    Console.WriteLine();
```

1.4->

Image1:



Euclidean distance : √(a-c)’ 2 + (b-d)’2

√ (936,0055-936)’2 + (523,9856-524)’2 = 0,01541

√ (936,0072-936)’2 + (758,3387-758)’2 = 0,33870

√ (1166,9822-1167)’2 + (761,3418-762)’2 = 0,65820

Image2:

```
For Image2 Homography Matrix

0,0041  0,0000  0,4850
-0,0001  0,0045  0,8745
0,0000  0,0000  0,0018

----------------Image2-------------------------

x = 200,0000      y = 5,0000
u = 737,0065    v = 492,7795
u = 737,0000    v = 493,0000
x = 199,9980    y = 5,0872

x = 200,0000      y = 100,0000
u = 735,1356    v = 731,6058
u = 735,0000    v = 731,0000
x = 199,9397    y = 99,7565

x = 300,0000      y = 100,0000
u = 968,7256    v = 727,0116
u = 969,0000    v = 728,0000
x = 300,1228    y = 100,3962

-----------------------------------------------
```

Euclidean distance : √(a-c)’ 2 + (b-d)’2

√ (737,0065-737)’2 + (492,7795-493)’2 = 0,2205

√ (735,1356-735)’2 + (731,6058-731)’2 = 0,6207

√ (968,7256-969)’2 + (727,0116-728)’2 = 0,2746

Image3:

```
For Image3 Homography Matrix

0,0036  0,0002  0,6421
0,0004  0,0039  0,7666
0,0000  0,0000  0,0013

----------------Image3-------------------------

x = 200,0000      y = 5,0000
u = 997,0107    v = 636,9068
u = 997,0000    v = 637,0000
x = 199,9967    y = 5,0342

x = 200,0000      y = 100,0000
u = 990,0641    v = 892,7973
u = 990,0000    v = 893,0000
x = 199,9758    y = 100,0780

x = 300,0000      y = 100,0000
u = 1227,8646    v = 907,4902
u = 1228,0000    v = 907,0000
x = 300,0485    y = 99,8087

-----------------------------------------------
```

Euclidean distance: **√(a-c)’ 2 + (b-d)’2**

√ (997,0107-997)’2 + (636,9068-637)’2 = 0,0931

√ (990,06417-990)’2 + (892,7973-893)’2 = 04943

√ (1227,8646-1228)’2 + (907,4902-907)’2 = 0,508

1.5->

$$\rho \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

1.4'te bulduğumuz H matrixlerine verilen x,y noktaları yukarıdaki eşitlikte yerine yazıldı, çıkan sonucu ise 1.satır ve 2.satırı 3. Satıra bölerek set edildi ve u,v ler bulundu.

Image1:

```
********** PART1 -5 **********

x = 7,5000        y = 5,5000
u = 481,8509      v = 517,3854

x = 6,3000        y = 3,3000
u = 478,9025      v = 511,8134

x = 0,1000        y = 0,1000
u = 464,0044      v = 503,5176
```

Image2:

```
********** PART1 -5 **********

x = 7,5000        y = 5,5000
u = 289,7727      v = 504,6072

x = 6,3000        y = 3,3000
u = 286,9876      v = 499,1518

x = 0,1000        y = 0,1000
u = 272,6644      v = 491,4597
```

Image3:

```
********** PART1 -5 **********

x = 7,5000       y = 5,5000
u = 507,5611     v = 599,5212

x = 6,3000       y = 3,3000
u = 504,3605     v = 593,0152

x = 0,1000       y = 0,1000
u = 487,9602     v = 582,5954
```

1.6->

$$\rho \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

1.4'te bulduğumuz H matrixlerine verilen u,v noktaları yukarıdaki eşitlikte yerine yazıldı, daha sonra H nin tersi ile u,v den oluşan matrix çarpılarak , 1.satır ve 2.satırı 3. Satıra bölerek set edildi ve x,y ler bulundu.

Image1:

```
********** PART1 -6 **********

u = 500,0000      v = 400,0000
x = 15,9287     y = -41,2247


u = 86,0000     v = 167,0000
x = -153,2845     y = -128,7676

u = 10,0000     v = 10,0000
x = -182,2098     y = -188,2648
```

Image2:

```
********** PART1 -6 **********

u = 500,0000    v = 400,0000
x = 98,2622    y = -33,9704


u = 86,0000    v = 167,0000
x = -79,9491    y = -130,2663

u = 10,0000    v = 10,0000
x = -112,2051    y = -192,7032
```

Image3:

```
********** PART1 -6 **********

u = 500,0000    v = 400,0000
x = 5,2093    y = -63,5441


u = 86,0000    v = 167,0000
x = -143,4302    y = -127,6136

u = 10,0000    v = 10,0000
x = -167,8469    y = -175,3013
```

PART2:

~ Bu partta silindiri belirtilen konuma yerleştirme işlemini iki şekilde yapabiliriz.

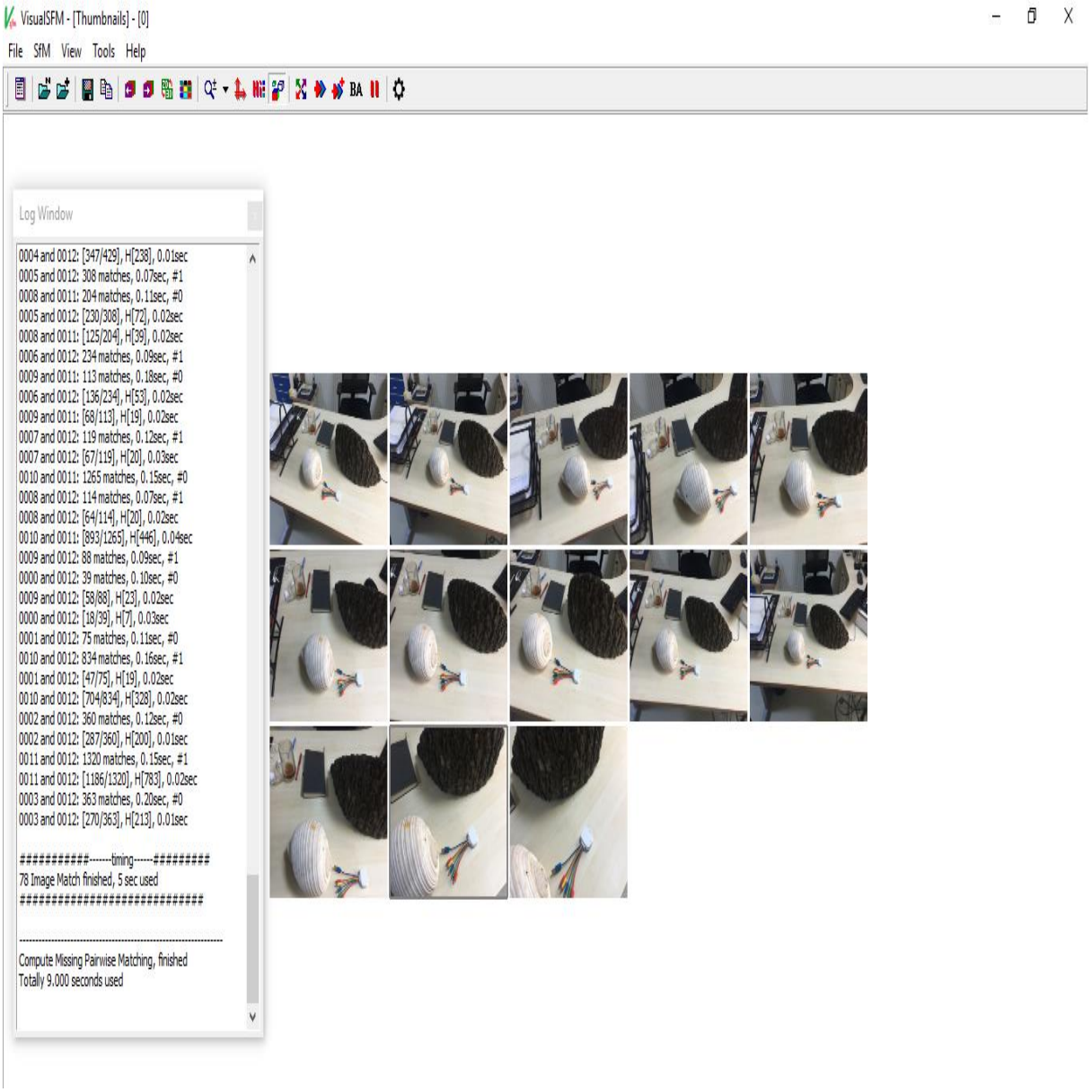
1- **Imageler üzerinden giderek ilk image source kabul edilir,daha sonra diğer tüm imagelerin H matrixi ilk image göre hesaplanır.İlk imageden silindirin oluşturduğu merkezin x,y leri verilerek her image için target noktalar bulunur.Bulunan target noktaları arasındaki uzaklıkları orantılayarak scale elde edilirken , bir birine olan konumlarından ise rotate elde edilir(her image için uygulanır).Son olarak ise bu bilgileri kullanarak 3 boyutlu cismimiz için linelar çizilir.**

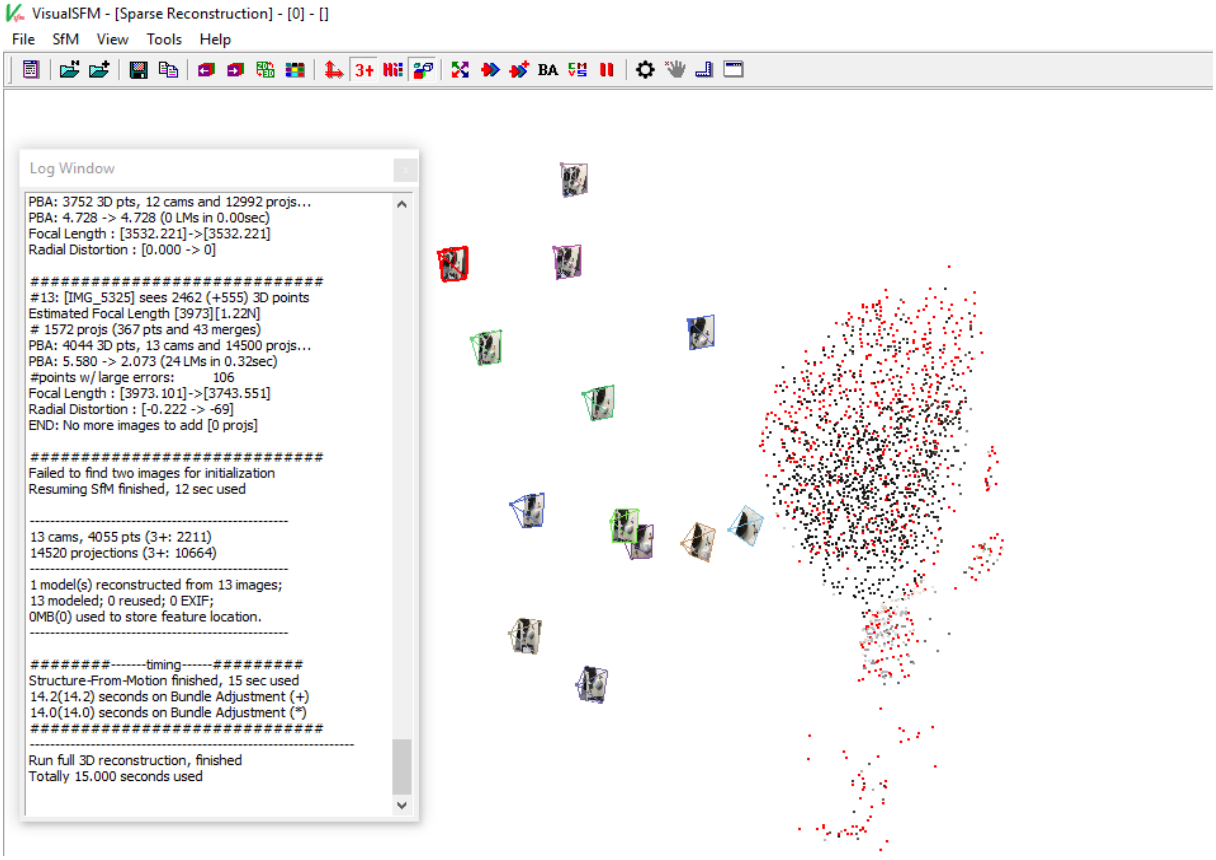
Kodsal olarak scale ve rotate hesaplama :

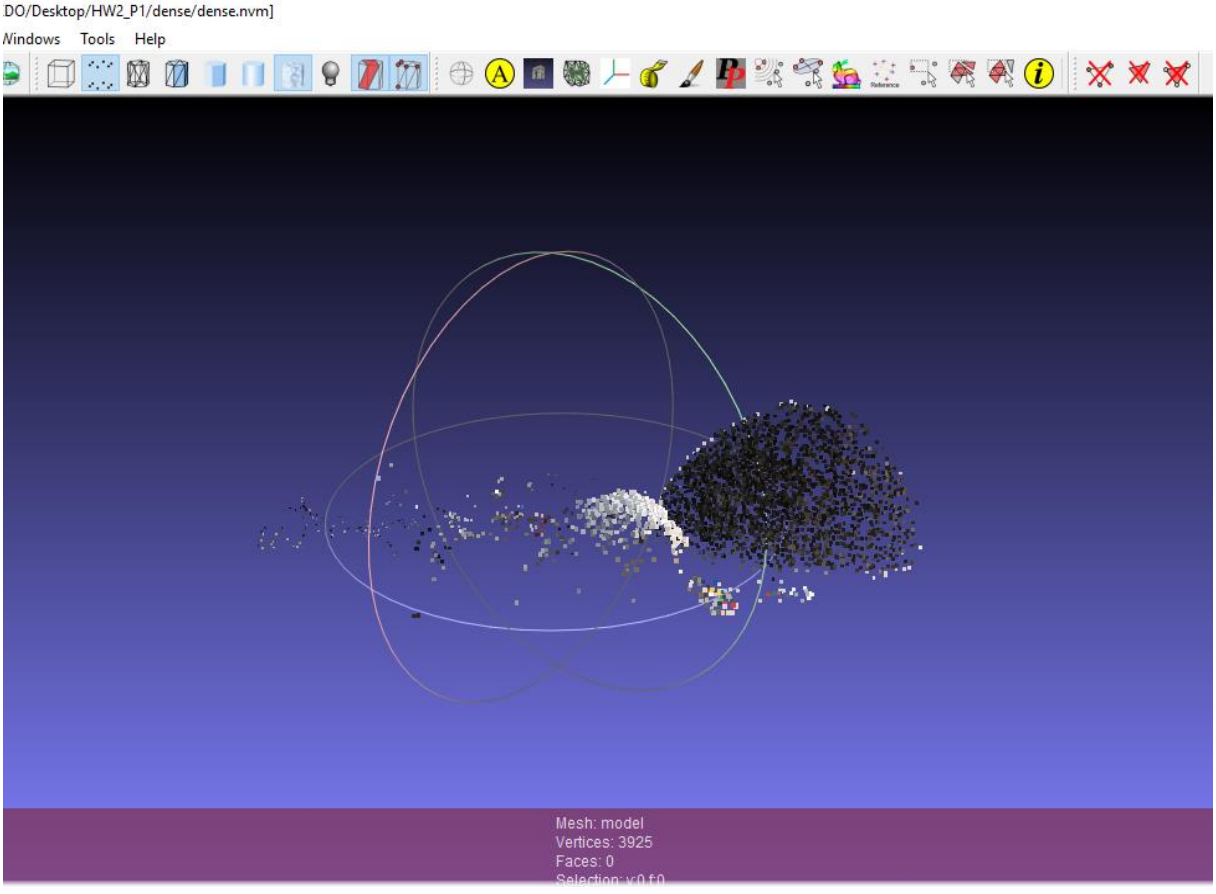
```
Console.WriteLine("Scale :  " + getDistance(source_img1, source_img1)/getDistance(target1,target2));

var dot = point2_img1[0] * (result2_img1[0] - result1_img1[0]) + point2_img1[1] * (result2_img1[1] - result1_img1[1]);

 var rotateAngle = Math.Acos(dot /getDistance(point2_img1, new double[] { 0, 0 }) * getDistance(result2_img1, result1_img1)));
```

2- **Visual SFM çalıştırılarak verilen imagelerden, cameralar ve sahne modellenir.Sahne virtual olarak alınır,silindirin konumu belirlenir.3 boyutlu noktalar işaretlenerek her bir camera için uygulanır.**

Visual SFM:




VisualSFM - [Sparse Reconstruction] - [0] - []
File SfM View Tools Help
Log Window
PBA: 3752 3D pts, 12 cams and 12992 projs...
PBA: 4.728 -> 4.728 (0 LMs in 0.00sec)
Focal Length : [3532.221]->[3532.221]
Radial Distortion : [0.000 -> 0]
##############################
#13: [IMG_5325] sees 2462 (+555) 3D points
Estimated Focal Length [3973][1.22N]
1572 projs (367 pts and 43 merges)
PBA: 4044 3D pts, 13 cams and 14500 projs...
PBA: 5.580 -> 2.073 (24 LMs in 0.32sec)
#points w/ large errors: 106
Focal Length : [3973.101]->[3743.551]
Radial Distortion : [-0.222 -> -69]
END: No more images to add [0 projs]
##############################
Failed to find two images for initialization
Resuming SfM finished, 12 sec used
13 cams, 4055 pts (3+: 2211)
14520 projections (3+: 10664)
1 model(s) reconstructed from 13 images;
13 modeled; 0 reused; 0 EXIF;
0MB(0) used to store feature location.
########-------timing------#########
Structure-From-Motion finished, 15 sec used
14.2(14.2) seconds on Bundle Adjustment (+)
14.0(14.0) seconds on Bundle Adjustment (*)
##############################
Run full 3D reconstruction, finished
Totally 15.000 seconds used


DO/Desktop/HW2_P1/dense/dense.nvm]
Windows Tools Help
Mesh: model
Vertices: 3925
Faces: 0