Haroun Habeeb **2013CS10225**

COL726: Assignment 2

Solutions to Q5, Q6 in minor 1

Implementations were done using python's numpy.

I use column vectors.

Q5:

We write some quick code to compute null space & range space of A in terms of Q, R.

Basis of null(A)

[[-0.15245872 0.52607636]

[-0.18883588 -0.81507117]

[0.83504791 0.05191325]

[-0.49375331 0.23708156]]

Basis of null(R)

[[0.07800837 0.542139]

[0.3000754 -0.780996]

[-0.83417589 -0.06442499]

[0.45609213 0.303282]]

Basis of Rangespace(A)

[[-0.4284124 0.71865348]

[-0.47437252 0.27380781]

[-0.52033264 -0.17103786]

[-0.56629275 -0.61588352]]

Basis of Rangespace(Q)

[[0. 1. -0. -0.]

[-0.70436073 0. 0.23624977 0.66937435]

[0.70436073 0. 0.34957871 0.61779502]

[0.08804509 0. -0.90663155 0.41263457]]

Basis of Rangespace(R)

[[-0.4284124 -0.71865348]

[-0.47437252 -0.27380781]

[-0.52033264 0.17103786]

[-0.56629275 0.61588352]]

[null(A) | null(R)]

[[-0.15245872 -0.18883588 0.83504791 -0.49375331]

[0.52607636 -0.81507117 0.05191325 0.23708156]

[0.07800837 0.3000754 -0.83417589 0.45609213]

[0.542139   -0.780996   -0.06442499 0.303282]]

Rank of [null(A) | null(R)] = 2 … demonstrates that null space of A = null space of R

Q6:

We write code that takes in 4 points, their correspondences and returns two functions – one which projects a point in 3D onto the two orthographic projects, and the other which does the reverse. We verify that the function works.

I'm not sure what else to write in the report. What I do know I can write is cases when the system fails.

Our system for recovering A_k is to first solve the linear system of equations in which $L(A_k - A_0) = \begin{bmatrix} a_k - a_0 \\ a'_k - a'_0 \end{bmatrix}$ where L is the big matrix given in the question. Notice that the solutions to this system of equations is in the basis $\{A_i - A_0\}_{i=1}^{3}$. To convert the solution back to $\mathbb{R}^3$, we must have a basis change matrix.

The reconstruction method can fail under two circumstances:

Case 1: The system of equations $L(A_k - A_0) = \begin{bmatrix} a_k - a_0 \\ a'_k - a'_0 \end{bmatrix}$ doesn't have a unique solution.

Case 2: The basis change is non-invertible.

Case 2 arises if the vectors $\{A_i\}_{i=0}^{3}$ are co-planar. For example,

```
A0 = np.matrix([0,0,0]).T
a0 = np.matrix([0,0]).T
ap0 = np.matrix([0,0]).T

A1 = np.matrix([2,1,0]).T
a1 = np.matrix([2,1]).T
ap1 = np.matrix([1,0]).T

A2 = np.matrix([1,2,0]).T
a2 = np.matrix([1,2]).T
ap2 = np.matrix([2,0]).T

A3 = np.matrix([3,3,0]).T
a3 = np.matrix([3,3]).T
ap3 = np.matrix([3,0]).T
```

Produces a basis change vector is is not invertible (singular). Note that $A_3 - A_0 = A_1 + A_2 - 2A_0$.

```
numpy.linalg.linalg.LinAlgError: Singular matrix
```

Case 1 can arise either if the matrix L is not full rank, or is inconsistent. The physical interpretation of not full rank is that the two viewing planes are parallel to each other.

```
A0 = np.matrix([0,0,0]).T
a0 = np.matrix([0,0]).T
ap0 = np.matrix([0,0]).T

A1 = np.matrix([2,1,0]).T
a1 = np.matrix([2,1]).T
ap1 = np.matrix([2,1]).T

A2 = np.matrix([1,2,0]).T
a2 = np.matrix([1,2]).T
ap2 = np.matrix([1,2]).T

A3 = np.matrix([3,3,0]).T
a3 = np.matrix([3,3]).T
ap3 = np.matrix([3,3]).T
```

Produces the error that L is non-invertible. (Note: Inverse is computed by augmenting L with a dimension of zeros).

The other case, where the system of equations is inconsistent, can generated by measurement errors. For example,

```
A0 = np.matrix([0,0,0]).T
a0 = np.matrix([0,0]).T
ap0 = np.matrix([0,0]).T

A1 = np.matrix([2,1,0]).T
a1 = np.matrix([2,1]).T
ap1 = np.matrix([1,0]).T

A2 = np.matrix([1,2,0]).T
a2 = np.matrix([1,2]).T
ap2 = np.matrix([1,0]).T

A3 = np.matrix([2,2,-2]).T
a3 = np.matrix([2,2]).T
ap3 = np.matrix([2,-2]).T
```

also produces no solution.